Tell us about your PDF experience.

Get started

Article • 06/24/2021

Build innovative and differentiated devices with Windows 10 and Windows 11. Windows runs on a broad set of devices—from desktops, notebooks, and Internet of Things (IoT) devices. The operating system's common core works across platforms with 80-inch screens or devices with no screens at all.

You can create devices to use touch/pen, mouse/keyboard, controller/gesture—or you can build them to switch between input types.

Start building devices

Expand table

Build a Windows PC	Build drivers for a Windows device
Download the Windows ADK to start building Windows PCs	Download the WDK to start building drivers
What's new in Windows 11	What's new in driver development
Design your deployment	Develop Universal Windows Drivers
Customize your Windows PC	Sign drivers
Create an end-to-end deployment	Test drivers

Tip

See **Kits and tools overview** to learn how our kits and tools enable you to work with Windows.

Design hardware with the latest features

Building a great Windows 10 or Windows 11 device starts with selecting the right components. Understanding the difference between minimum and optimal hardware requirements will ensure the best usability and performance. Our guidelines explain the requirements to implement Windows features like Cortana, Windows Hello, Touch, Windows Pen, and Continuum.

- Hardware component guidelines

Customize Windows images to reflect your brand

For desktop PCs, you can use your existing settings file (Unattend.xml) to add settings during Windows installation.

- Build a Windows Setup unattend file

Develop Universal Windows Drivers

Learn fundamental concepts about drivers.

- Concepts for all driver developers

Create a single driver that runs across multiple device types, from embedded systems to tablets and desktop PCs. UMDF and KMDF templates are included in Visual Studio to help you get started.

- Get started with Universal Windows drivers

Test system components for compatibility and performance

Expand table

Write and run test automations with the Test Authoring and Execution Framework (TAEF). Share your tests across disciplines and teams. Get started with the Test Authoring and Execution Framework (TAEF)	Test your hardware with the Windows Hardware Lab Kit. Get started with the Windows Hardware Lab Kit	Analyze system and application performance using the Windows Performance Toolkit. Get started with the Windows Performance step-by-step guides

Manufacturing – putting it all together

Expand table

For desktop PCs, learn strategies to build a set of images for specific markets to meet different customers' needs. Build and deploy desktop devices	Learn about deploying Windows 10 in S mode, a configuration of Windows 10 Pro that offers a familiar, productive Windows experience that's streamlined for security and performance. Windows 10 S manufacturing overview	Build IoT Core devices, applying apps, drivers, and settings to new devices. Build and deploy IoT Core devices

Design devices

Article • 06/24/2021

Building a great Windows 10 or Windows 11 device starts with selecting the right components. Understanding the difference between minimum and optimal hardware requirements will ensure the best usability and performance. To take advantage of the latest features like Cortana and Continuum, your hardware must meet specific requirements to ensure an optimal device experience.

Form factors

Windows 10 and Windows 11 support a wide range of devices—from tablets to laptops, to All-in-one or desktop devices. See Form factors to choose your device and learn more info about creating great Windows devices.

Design guidelines

Review the minimum hardware requirements

The minimum hardware requirements specification defines the minimum hardware requirements necessary to boot and run Windows 10 and Windows 11, update and service Windows 10 and Windows 11, and provide a baseline user experience that is comparable with similar devices and computers.

Minimum hardware requirements enable OEMs, ODMs, SoC vendors, and other component vendors to make early design decisions for devices and computers that will run Windows 10 or Windows 11.

Review the minimum hardware requirements

Plan for a rich user experience

Build devices that take full advantage of the latest features. To get started learn how the feature works and then choose the right hardware.

Learn more about the latest Windows features

Select the right components

Building a great device for Windows means with selecting the right components. Microsoft makes recommendations to help you put together the best system for running Windows 10 and Windows 11.

Learn more about individual component guidelines

Test it to make sure it's compatible and reliable

We provide tests that help you deliver systems, software and hardware products that are compatible with Windows and run reliably on Windows 10 and Windows 11. We also give you the opportunity to promote your product on Microsoft's Certified Products List and with the Windows logo.

Learn more about the Windows Hardware Compatibility Program

What's new in Windows 11

Article • 05/25/2022

Welcome to Windows 11! Here are some resources to get you acquainted with the new features and changes introduced for Windows 11.

New in manufacturing

Topic	Description
Validation OS	Validation OS is a lightweight OS image based on Windows 11 configured to support early hardware and factory line bring-up. It boots to a Command Line interface to increase reliability on the factory line.
Languages overview	Languages • More easily add and manage languages and Features on demand with the new Languages and Optional Features ISO. This ISO combines the previously separate Features on demand and language packs ISOs, and can be used as a FOD and Language pack repository. • Only LIP languages distributed as .cabs can be used in manufacturing. 5 LIP languages are now available as .cabs.
Features on Demand	The Features on Demand ISO has been replaced with the Languages and Optional Features ISO. This ISO includes FOD packages and languages packs.
Manage inbox network drivers	Inbox PCI ethernet and WiFi drivers are now Features on Demand. You can easily remove these network driver FODs to reduce your disk footprint.
Using deployment tools with Windows 11 images	When using deployment tools to work Windows 11 images, the Name you expect to use with images might be different than you expect. This topic contains an overview of the issue as well as some workarounds.
Windows Setup Command Line Options	New `/EULA` command line options lets you accept the EULA when you launch setup.exe

New customization options

Topic	Description

Topic	Description
Customize the Start menu	Windows 11 has a new way to configure the apps that are pinned to the Start menu.
Configure the Taskbar	Updated Taskbar topic to reflect changes to the Taskbar.
Desktop background and themes	Gives guidance on how to create a custom wallpaper that is optimized for the Windows 11 experience.
Unattend settings	You can now use Unattend to disable the startup sound.
Customize OOBE	Updated docs include new OOBE screen flow, information about the new quiet period, the new `<ShowPhoneNumber>` element for oobe.xml, and updated design guidance for creating a EULA.
Settings for better tablet experiences	Covers how to enable optimized experiences when tablets and 2-in-1 devices are in tablet posture.

New in design

Topic	Description
Hypervisor-protected Code Integrity enablement	New installations on compatible systems have memory integrity turned on by default, though device manufacturers and end users have the ultimate control of whether the feature is enabled.
Docking and multi-monitor experiences	Covers the improved experience for users who work with multiple apps at a time. Provides an overview of docking/redocking, snap groups, and guidance for docking stations.

Windows minimum hardware requirements

Article • 09/01/2021

The specification defines the Windows minimum hardware requirements necessary to:

- Boot and run Windows
- Update and service Windows
- Provide a baseline user experience that is comparable with similar devices and computers

The goal of this specification is to enable OEMs, ODMs, Silicon, and other component vendors to make early design decisions for devices and computers that will run Windows.

This specification does not provide compatibility and certification requirements for component and device that run Windows or implementation guidance for exceptional user experience. For Windows Hardware Compatibility Program requirements, policies, and Windows Lab Kits (HLK), refer to https://aka.ms/WHCP

Download minimum hardware requirements

- Windows 11
- Windows 10

Windows Processor Requirements

Article • 11/01/2024

If you are a customer seeking to determine Windows 11 eligibility for your device, please use PC Health Check. For Windows support life cycles please see Windows 11 Home and Pro - Microsoft Lifecycle.

This specification details the processors that Original Equipment Manufacturers (OEM) can use to manufacture new PCs that include Windows Products (including Custom Images). Updates to this specification may be released in the future as requirements change.

For each listed edition, OEMs must use only the processors listed, as specified in the tables below. The requirements below apply whenever the edition below is pre-installed or provided on external media, including as downgrade or down edition software.

For clarity, OEMs must also meet all processor and other requirements specified in Minimum Hardware Requirements for Windows, available at Windows minimum hardware requirements.

If after the inclusion of a processor series in this specification ("Listed Processor"), a processor becomes commercially available that uses the same naming convention or identifier as a Listed Processor but has additional or different features or functionality ("New Processor"), OEMs must not use New Processor for Customer Systems without Microsoft's prior written permission. If an OEM believes a processor has been omitted from this list, please contact your Microsoft OEM or ODM Account Manager.

Some product editions or edition/processor configurations listed below may have no or limited support. Information on support is available at Microsoft Support Policy and Microsoft Lifecycle FAQ. Customers who have specific hardware support questions should contact the OEM of their PC.

Changes to the processors listed do not indicate or impact a customer's existing Windows support and are intended for OEMs to determine processors which may be used in new Windows devices.

Windows Client Edition Processors

Expand table

Windows Edition	AMD Processors	Intel Processors	Qualcomm Processors

Windows 7 and earlier editions	Supported AMD Processors	Supported Intel Processors	N/A
Windows 8.1	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 Enterprise LTSB 1507	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 1511	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 1607	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 Enterprise LTSB 1607	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 1703	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 1709	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 1803	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 1809	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 Enterprise LTSC 1809	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 1903	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 1909	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 2004	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 20H2	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 21H1	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 10 21H2	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors

Windows 10 Enterprise LTSC 2021	Supported AMD Processors	Supported Intel Processors	N/A
Windows 10 22H2	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 11 version 21H2	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 11 version 22H2 & 23H2	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors
Windows 11 version 24H2	Supported AMD Processors	Supported Intel Processors	Supported Qualcomm Processors

Windows IoT Processors

This table provides reference to additional processor support and hardware requirements for Windows IoT Enterprise editions and Windows 10 IoT Core.

Expand table

Windows IoT Edition	Supported Processors
Windows 10 IoT Enterprise LTSC	See Windows 10 IoT Enterprise LTSC supported processors
Windows 11 IoT Enterprise	See Windows 11 IoT Enterprise supported processors
Windows 10 IoT Enterprise	See Windows 10 IoT Enterprise supported processors
Windows 10 IoT Core	See Windows 10 IoT Core supported processors

Windows Server Processors

Expand table

Windows Edition	Intel Processors	AMD Processors	Hygon Processors[7]
Windows Server 2012 R2[4]	Up through the following 7th Generation Intel Processors (Intel Core i3-7xxx/Celeron/Pentium; Xeon E3 v6); Xeon SP 32xx, 42xx, 52xx, 62xx, and 82xx; Xeon D 15xx; and Atom C33xx	Up through the following AMD 7th generation processors (AMD A-Series Ax-9xxx & E-Series Ex-9xxx & FX-9xxx), AMD Ryzen Family, AMD EPYC	N/A

		7xx1, AMD EPYC 7xx2, and AMD EPYC 7xx3	
Windows Server 2016[5]	Up through the following 9th Generation Intel Processors (Core i3-9xxx, Pentium G5xxx, Celeron G49xx); Xeon E22xx; Xeon SP 32xx, 43xx, 53xx, 63xx, and 83xx; Xeon D 21xx; and Atom C33xx	Up through the following AMD 7th generation processors (AMD A-Series Ax-9xxx & E-Series Ex-9xxx & FX-9xxx), AMD Ryzen Family, AMD EPYC 7xx1, AMD EPYC 7xx2, and AMD EPYC 7xx3	N/A
Windows Server 2019[6]	Up through the following 13th Generation Intel Processors (Core i3-13xxx, Pentium G5xxx, G7400 and G7400T, Celeron G49xx); Xeon E23xx; First through Fifth Gen Xeon SP Processors (x1xx, x2xx, x3xx, x4xx, x5xx); Xeon E 21xx through 24xx; Xeon D 15xx, 17xx, 21xx, and 27xx; and Atom C33xx	Up through the following AMD 7th generation processors (AMD A-Series Ax-9xxx & E-Series Ex-9xxx & FX-9xxx), AMD Ryzen Family, AMD EPYC 7xx1, AMD EPYC 7xx2, AMD EPYC 7xx3, AMD EPYC 8xx4, AMD EPYC 9xx4, and AMD EPYC 9xx5	Hygon C86 7xxx
Windows Server 2022	Up through the following 13th Generation Intel Processors (Core i3-13xxx, Pentium G5xxx, G7400 and G7400T, Celeron G49xx); First through Fifth Gen Xeon SP processors (x1xx, x2xx, x3xx, x4xx, x5xx); Xeon 6 SP processors (67xxE); Xeon E 23xx and 24xx; Xeon D 17xx, 21xx, and 27xx; and Atom C33xx	Up through the following AMD 7th generation processors (AMD A-Series Ax-9xxx & E-Series Ex-9xxx & FX-9xxx), AMD Ryzen Family, AMD EPYC 7xx1, AMD EPYC 7xx2, AMD EPYC 7xx3, AMD EPYC 4xx4, AMD EPYC 8xx4, AMD EPYC 9xx4, and AMD EPYC 9xx5	N/A
Windows Server 2025	Second through Fifth Gen Xeon SP processors (x2xx, x3xx, x4xx, x5xx); Xeon 6 SP Processors (67xxE); Xeon E 23xx and 24xx; Xeon D 17xx, 18xx, 21xx, 27xx, 28xx; Pentium G7400 and G7400T	AMD EPYC 7xx2, AMD EPYC 7xx3, AMD EPYC 4xx4, AMD EPYC 8xx4, AMD EPYC 9xx4, and AMD EPYC 9xx5	N/A

[4] The processor list for Windows Server 2012 R2 is final. New system submissions are no longer accepted for certification.

[5] The processor list for Windows Server 2016 is final. New system submissions are no longer accepted for certification.

[6] The processor list for Windows Server 2019 is final. New system submissions are no longer accepted for certification.

[7] China Market Only

NOTE: The list of supported processors above does not in itself determine Microsoft support for Windows Server. The listing is a prerequisite for system certification. Only systems based on the above approved processors can be certified for Windows Server. Unless otherwise noted, Microsoft will continue to evaluate the processor list for a given OS release and update the list as new appropriate processors are available in market.

Additional Resources

Windows Server support and installation instructions for the AMD Rome family of processors

Windows Server 2016 System Requirements

Comparison of Standard and Datacenter editions of Windows Server 2016

Windows 10 system requirements

Download the Windows Server 2016 licensing datasheet

Windows 7 and earlier editions Supported AMD Processors

Article • 11/10/2021

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics

Manufacturer	Brand	Model
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics

Manufacturer	Brand	Model
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10-8700B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU

Windows 7 and earlier editions Supported Intel Processors

Article • 11/10/2021

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3955U
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E

Manufacturer	Brand	Model
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U

Manufacturer	Brand	Model
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R

Manufacturer	Brand	Model
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820X
Intel®	Core™	i7-9800X

Manufacturer	Brand	Model
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y

Manufacturer	Brand	Model
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540

Manufacturer	Brand	Model
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1235L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T

Manufacturer	Brand	Model
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 8.1 Supported AMD Processors

Article • 11/10/2021

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics

Manufacturer	Brand	Model
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics

Manufacturer	Brand	Model
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10-8700B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU

Windows 8.1 Supported Intel Processors

Article • 11/10/2021

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3955U
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H

Manufacturer	Brand	Model
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ

Manufacturer	Brand	Model
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820X
Intel®	Core™	i7-9800X
Intel®	Core™	i9-7900X

Manufacturer	Brand	Model
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507

Manufacturer	Brand	Model
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541

Manufacturer	Brand	Model
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123

Manufacturer	Brand	Model
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 1511 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320

Manufacturer	Brand	Model
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
	Series	
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B

Windows 10 1511 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE

Manufacturer	Brand	Model
Intel®	Celeron®	G3950
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U

Manufacturer	Brand	Model
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U

Manufacturer	Brand	Model
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X

Manufacturer	Brand	Model
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-9800X
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a

Manufacturer	Brand	Model
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620

Manufacturer	Brand	Model
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571

Manufacturer	Brand	Model
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 1607 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320

Manufacturer	Brand	Model
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
	Series	
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B

Windows 10 1607 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE

Manufacturer	Brand	Model
Intel®	Celeron®	G3950
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U

Manufacturer	Brand	Model
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U

Manufacturer	Brand	Model
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X

Manufacturer	Brand	Model
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-9800X
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a

Manufacturer	Brand	Model
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620

Manufacturer	Brand	Model
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571

Manufacturer	Brand	Model
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 1703 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler

Manufacturer	Brand	Model
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2300U

Manufacturer	Brand	Model
	PRO	
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X

Manufacturer	Brand	Model
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X

Windows 10 1703 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE

Manufacturer	Brand	Model
Intel®	Celeron®	G3950
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U

Manufacturer	Brand	Model
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U

Manufacturer	Brand	Model
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X

Manufacturer	Brand	Model
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-9800X
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a

Manufacturer	Brand	Model
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620

Manufacturer	Brand	Model
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571

Manufacturer	Brand	Model
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 1709 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3 PRO	1200

Manufacturer	Brand	Model
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X

Manufacturer	Brand	Model
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX

Windows 10 1709 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	4205U
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920

Manufacturer	Brand	Model
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	J4005
Intel®	Celeron®	J4105
Intel®	Celeron®	N4000
Intel®	Celeron®	N4100
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E

Manufacturer	Brand	Model
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100

Manufacturer	Brand	Model
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U

Manufacturer	Brand	Model
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y

Manufacturer	Brand	Model
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i7+8700
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ

Manufacturer	Brand	Model
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U

Manufacturer	Brand	Model
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K

Manufacturer	Brand	Model
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9800X
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51

Manufacturer	Brand	Model
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y

Manufacturer	Brand	Model
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver N5000
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531

Manufacturer	Brand	Model
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT

Manufacturer	Brand	Model
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME

Manufacturer	Brand	Model
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123

Manufacturer	Brand	Model
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 1709 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	835

Windows 10 1803 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3 PRO	1200

Manufacturer	Brand	Model
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X

Manufacturer	Brand	Model
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX

Windows 10 1803 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	4205U
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920

Manufacturer	Brand	Model
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	J4005
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105
Intel®	Celeron®	J4125
Intel®	Celeron®	N4000
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100
Intel®	Celeron®	N4120
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U

Manufacturer	Brand	Model
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300

Manufacturer	Brand	Model
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF

Manufacturer	Brand	Model
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ

Manufacturer	Brand	Model
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K

Manufacturer	Brand	Model
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600

Manufacturer	Brand	Model
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i7+8700
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ

Manufacturer	Brand	Model
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ

Manufacturer	Brand	Model
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U

Manufacturer	Brand	Model
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK

Manufacturer	Brand	Model
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71

Manufacturer	Brand	Model
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y

Manufacturer	Brand	Model
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000
Intel®	Pentium®	Silver N5030
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531

Manufacturer	Brand	Model
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT

Manufacturer	Brand	Model
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME

Manufacturer	Brand	Model
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123

Manufacturer	Brand	Model
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 1803 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	835
Qualcomm®	Snapdragon™	850

Windows 10 1809 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3 PRO	1200

Manufacturer	Brand	Model
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X

Manufacturer	Brand	Model
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX

Windows 10 1809 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	4205U
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE

Manufacturer	Brand	Model
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925
Intel®	Celeron®	J4005
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105

Manufacturer	Brand	Model
Intel®	Celeron®	J4125
Intel®	Celeron®	N4000
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100
Intel®	Celeron®	N4120
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4
Intel®	Core™	i3-1005G1
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-5005U

Manufacturer	Brand	Model
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE

Manufacturer	Brand	Model
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300

Manufacturer	Brand	Model
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10200H
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T

Manufacturer	Brand	Model
Intel®	Core™	i5-10500TE
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P

Manufacturer	Brand	Model
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600

Manufacturer	Brand	Model
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T

Manufacturer	Brand	Model
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i7+8700
Intel®	Core™	i7-10510U
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700

Manufacturer	Brand	Model
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U

Manufacturer	Brand	Model
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ

Manufacturer	Brand	Model
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H

Manufacturer	Brand	Model
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK

Manufacturer	Brand	Model
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57

Manufacturer	Brand	Model
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	6805
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T

Manufacturer	Brand	Model
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000
Intel®	Pentium®	Silver N5030
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539

Manufacturer	Brand	Model
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G

Manufacturer	Brand	Model
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE

Manufacturer	Brand	Model
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L
Intel®	Xeon®	Gold 6240R
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold Gold 5218
Intel®	Xeon®	Gold Gold 5318Y
Intel®	Xeon®	Gold Gold 6238R
Intel®	Xeon®	Gold6246
Intel®	Xeon®	Goldv 6234
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-2223

Manufacturer	Brand	Model
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M

Windows 10 1809 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	850

Windows 10 1903 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3100

Manufacturer	Brand	Model
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE (OEM Only)
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3300X
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600

Manufacturer	Brand	Model
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE (OEM Only)
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor (OEM Only)
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	3600

Manufacturer	Brand	Model
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 9	3900 Processor (OEM Only)
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X

Manufacturer	Brand	Model
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 1903 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Atom®	x6200FE
Intel®	Atom®	x6211E
Intel®	Atom®	x6212RE
Intel®	Atom®	x6413E
Intel®	Atom®	x6414RE
Intel®	Atom®	x6425E
Intel®	Atom®	x6425RE
Intel®	Atom®	x6427FE
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	4205U

Manufacturer	Brand	Model
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	6305
Intel®	Celeron®	6305E
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900

Manufacturer	Brand	Model
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925
Intel®	Celeron®	J4005
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105
Intel®	Celeron®	J4125
Intel®	Celeron®	J6412
Intel®	Celeron®	J6413
Intel®	Celeron®	N4000
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100
Intel®	Celeron®	N4120
Intel®	Celeron®	N4500
Intel®	Celeron®	N4505
Intel®	Celeron®	N5100
Intel®	Celeron®	N5105
Intel®	Celeron®	N6210
Intel®	Celeron®	N6211
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4
Intel®	Core™	i3-1005G1

Manufacturer	Brand	Model
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-1110G4
Intel®	Core™	i3-1115G4
Intel®	Core™	i3-1115G4E
Intel®	Core™	i3-1115GRE
Intel®	Core™	i3-1120G4
Intel®	Core™	i3-1125G4
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U

Manufacturer	Brand	Model
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U

Manufacturer	Brand	Model
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320

Manufacturer	Brand	Model
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i3-L13G4
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10200H
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T
Intel®	Core™	i5-10500TE

Manufacturer	Brand	Model
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T
Intel®	Core™	i5-11300H
Intel®	Core™	i5-1130G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-11400
Intel®	Core™	i5-11400F
Intel®	Core™	i5-11400T
Intel®	Core™	i5-1140G7
Intel®	Core™	i5-1145G7
Intel®	Core™	i5-1145G7E
Intel®	Core™	i5-1145GRE
Intel®	Core™	i5-11500
Intel®	Core™	i5-11500T
Intel®	Core™	i5-11600
Intel®	Core™	i5-11600K
Intel®	Core™	i5-11600KF
Intel®	Core™	i5-11600T
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U

Manufacturer	Brand	Model
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T

Manufacturer	Brand	Model
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U

Manufacturer	Brand	Model
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500

Manufacturer	Brand	Model
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i5-L16G7
Intel®	Core™	i7+8700
Intel®	Core™	i7-10510U
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U

Manufacturer	Brand	Model
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-11370H
Intel®	Core™	i7-11375H
Intel®	Core™	i7-1160G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-11700
Intel®	Core™	i7-11700F
Intel®	Core™	i7-11700K
Intel®	Core™	i7-11700KF
Intel®	Core™	i7-11700T
Intel®	Core™	i7-1180G7
Intel®	Core™	i7-1185G7
Intel®	Core™	i7-1185G7E
Intel®	Core™	i7-1185GRE
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C

Manufacturer	Brand	Model
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U

Manufacturer	Brand	Model
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700

Manufacturer	Brand	Model
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T

Manufacturer	Brand	Model
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-11900
Intel®	Core™	i9-11900F
Intel®	Core™	i9-11900K
Intel®	Core™	i9-11900KF
Intel®	Core™	i9-11900T
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X

Manufacturer	Brand	Model
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	6805
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509

Manufacturer	Brand	Model
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold 7505
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	J6426
Intel®	Pentium®	N6415
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000
Intel®	Pentium®	Silver N5030
Intel®	Pentium®	Silver N6000
Intel®	Pentium®	Silver N6005
Intel®	Xeon®	Bronze 3104

Manufacturer	Brand	Model
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT

Manufacturer	Brand	Model
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G

Manufacturer	Brand	Model
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L
Intel®	Xeon®	Gold 6240R

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold Gold 5218
Intel®	Xeon®	Gold Gold 5318Y
Intel®	Xeon®	Gold Gold 6238R
Intel®	Xeon®	Gold6246
Intel®	Xeon®	Goldv 6234
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P

Manufacturer	Brand	Model
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-2223
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235

Manufacturer	Brand	Model
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M

Windows 10 1903 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	850
Qualcomm®	Snapdragon™	8cx

Windows 10 1909 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3100

Manufacturer	Brand	Model
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE (OEM Only)
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3300X
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600

Manufacturer	Brand	Model
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE (OEM Only)
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor (OEM Only)
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	3600

Manufacturer	Brand	Model
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 9	3900 Processor (OEM Only)
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X

Manufacturer	Brand	Model
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 1909 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Atom®	x6200FE
Intel®	Atom®	x6211E
Intel®	Atom®	x6212RE
Intel®	Atom®	x6413E
Intel®	Atom®	x6414RE
Intel®	Atom®	x6425E
Intel®	Atom®	x6425RE
Intel®	Atom®	x6427FE
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	4205U

Manufacturer	Brand	Model
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	6305
Intel®	Celeron®	6305E
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900

Manufacturer	Brand	Model
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925
Intel®	Celeron®	J4005
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105
Intel®	Celeron®	J4125
Intel®	Celeron®	J6412
Intel®	Celeron®	J6413
Intel®	Celeron®	N4000
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100
Intel®	Celeron®	N4120
Intel®	Celeron®	N4500
Intel®	Celeron®	N4505
Intel®	Celeron®	N5100
Intel®	Celeron®	N5105
Intel®	Celeron®	N6210
Intel®	Celeron®	N6211
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4
Intel®	Core™	i3-1005G1

Manufacturer	Brand	Model
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-1110G4
Intel®	Core™	i3-1115G4
Intel®	Core™	i3-1115G4E
Intel®	Core™	i3-1115GRE
Intel®	Core™	i3-1120G4
Intel®	Core™	i3-1125G4
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U

Manufacturer	Brand	Model
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U

Manufacturer	Brand	Model
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320

Manufacturer	Brand	Model
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i3-L13G4
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10200H
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T
Intel®	Core™	i5-10500TE

Manufacturer	Brand	Model
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T
Intel®	Core™	i5-11300H
Intel®	Core™	i5-1130G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-11400
Intel®	Core™	i5-11400F
Intel®	Core™	i5-11400T
Intel®	Core™	i5-1140G7
Intel®	Core™	i5-1145G7
Intel®	Core™	i5-1145G7E
Intel®	Core™	i5-1145GRE
Intel®	Core™	i5-11500
Intel®	Core™	i5-11500T
Intel®	Core™	i5-11600
Intel®	Core™	i5-11600K
Intel®	Core™	i5-11600KF
Intel®	Core™	i5-11600T
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U

Manufacturer	Brand	Model
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T

Manufacturer	Brand	Model
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U

Manufacturer	Brand	Model
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500

Manufacturer	Brand	Model
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i5-L16G7
Intel®	Core™	i7+8700
Intel®	Core™	i7-10510U
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U

Manufacturer	Brand	Model
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-11370H
Intel®	Core™	i7-11375H
Intel®	Core™	i7-1160G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-11700
Intel®	Core™	i7-11700F
Intel®	Core™	i7-11700K
Intel®	Core™	i7-11700KF
Intel®	Core™	i7-11700T
Intel®	Core™	i7-1180G7
Intel®	Core™	i7-1185G7
Intel®	Core™	i7-1185G7E
Intel®	Core™	i7-1185GRE
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C

Manufacturer	Brand	Model
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U

Manufacturer	Brand	Model
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700

Manufacturer	Brand	Model
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T

Manufacturer	Brand	Model
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-11900
Intel®	Core™	i9-11900F
Intel®	Core™	i9-11900K
Intel®	Core™	i9-11900KF
Intel®	Core™	i9-11900T
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X

Manufacturer	Brand	Model
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	6805
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509

Manufacturer	Brand	Model
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold 7505
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	J6426
Intel®	Pentium®	N6415
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000
Intel®	Pentium®	Silver N5030
Intel®	Pentium®	Silver N6000
Intel®	Pentium®	Silver N6005
Intel®	Xeon®	Bronze 3104

Manufacturer	Brand	Model
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT

Manufacturer	Brand	Model
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G

Manufacturer	Brand	Model
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L
Intel®	Xeon®	Gold 6240R

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold Gold 5218
Intel®	Xeon®	Gold Gold 5318Y
Intel®	Xeon®	Gold Gold 6238R
Intel®	Xeon®	Gold6246
Intel®	Xeon®	Goldv 6234
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P

Manufacturer	Brand	Model
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-2223
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235

Manufacturer	Brand	Model
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M

Windows 10 1909 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	850
Qualcomm®	Snapdragon™	8cx

Windows 10 2004 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3100

Manufacturer	Brand	Model
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE (OEM Only)
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3300X
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G (OEM Only)
AMD	Ryzen™ 3	4300GE (OEM Only)
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)

Manufacturer	Brand	Model
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE (OEM Only)
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor (OEM Only)
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G (OEM Only)
AMD	Ryzen™ 5	4600GE (OEM Only)
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5 PRO	1500

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H

Manufacturer	Brand	Model
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G (OEM Only)
AMD	Ryzen™ 7	4700GE (OEM Only)
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 9	3900 Processor (OEM Only)
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9 PRO	3900

Manufacturer	Brand	Model
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 2004 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Atom®	x6200FE
Intel®	Atom®	x6211E
Intel®	Atom®	x6212RE
Intel®	Atom®	x6413E
Intel®	Atom®	x6414RE
Intel®	Atom®	x6425E
Intel®	Atom®	x6425RE
Intel®	Atom®	x6427FE
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y
Intel®	Celeron®	4205U

Manufacturer	Brand	Model
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	6305
Intel®	Celeron®	6305E
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900

Manufacturer	Brand	Model
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925
Intel®	Celeron®	J4005
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105
Intel®	Celeron®	J4125
Intel®	Celeron®	J6412
Intel®	Celeron®	J6413
Intel®	Celeron®	N4000
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100
Intel®	Celeron®	N4120
Intel®	Celeron®	N4500
Intel®	Celeron®	N4505
Intel®	Celeron®	N5100
Intel®	Celeron®	N5105
Intel®	Celeron®	N6210
Intel®	Celeron®	N6211
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4
Intel®	Core™	i3-1005G1

Manufacturer	Brand	Model
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-1110G4
Intel®	Core™	i3-1115G4
Intel®	Core™	i3-1115G4E
Intel®	Core™	i3-1115GRE
Intel®	Core™	i3-1120G4
Intel®	Core™	i3-1125G4
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U

Manufacturer	Brand	Model
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U

Manufacturer	Brand	Model
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320

Manufacturer	Brand	Model
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i3-L13G4
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10200H
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T
Intel®	Core™	i5-10500TE

Manufacturer	Brand	Model
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T
Intel®	Core™	i5-11300H
Intel®	Core™	i5-1130G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-11400
Intel®	Core™	i5-11400F
Intel®	Core™	i5-11400T
Intel®	Core™	i5-1140G7
Intel®	Core™	i5-1145G7
Intel®	Core™	i5-1145G7E
Intel®	Core™	i5-1145GRE
Intel®	Core™	i5-11500
Intel®	Core™	i5-11500T
Intel®	Core™	i5-11600
Intel®	Core™	i5-11600K
Intel®	Core™	i5-11600KF
Intel®	Core™	i5-11600T
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U

Manufacturer	Brand	Model
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T

Manufacturer	Brand	Model
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U

Manufacturer	Brand	Model
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500

Manufacturer	Brand	Model
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i5-L16G7
Intel®	Core™	i7+8700
Intel®	Core™	i7-10510U
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U

Manufacturer	Brand	Model
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-11370H
Intel®	Core™	i7-11375H
Intel®	Core™	i7-1160G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-11700
Intel®	Core™	i7-11700F
Intel®	Core™	i7-11700K
Intel®	Core™	i7-11700KF
Intel®	Core™	i7-11700T
Intel®	Core™	i7-1180G7
Intel®	Core™	i7-1185G7
Intel®	Core™	i7-1185G7E
Intel®	Core™	i7-1185GRE
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C

Manufacturer	Brand	Model
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U

Manufacturer	Brand	Model
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700

Manufacturer	Brand	Model
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T

Manufacturer	Brand	Model
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-11900
Intel®	Core™	i9-11900F
Intel®	Core™	i9-11900K
Intel®	Core™	i9-11900KF
Intel®	Core™	i9-11900T
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X

Manufacturer	Brand	Model
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	6805
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509

Manufacturer	Brand	Model
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U
Intel®	Pentium®	Gold 4415Y
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold 7505
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	J6426
Intel®	Pentium®	N6415
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000
Intel®	Pentium®	Silver N5030
Intel®	Pentium®	Silver N6000
Intel®	Pentium®	Silver N6005
Intel®	Xeon®	Bronze 3104

Manufacturer	Brand	Model
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT

Manufacturer	Brand	Model
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G

Manufacturer	Brand	Model
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L
Intel®	Xeon®	Gold 6240R

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold Gold 5218
Intel®	Xeon®	Gold Gold 5318Y
Intel®	Xeon®	Gold Gold 6238R
Intel®	Xeon®	Gold6246
Intel®	Xeon®	Goldv 6234
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P

Manufacturer	Brand	Model
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-2223
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235

Manufacturer	Brand	Model
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M

Windows 10 2004 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	850
Qualcomm®	Snapdragon™	8cx

Windows 10 20H2 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3100

Manufacturer	Brand	Model
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE (OEM Only)
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3300X
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G (OEM Only)
AMD	Ryzen™ 3	4300GE (OEM Only)
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)

Manufacturer	Brand	Model
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE (OEM Only)
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor (OEM Only)
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G (OEM Only)
AMD	Ryzen™ 5	4600GE (OEM Only)
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5 PRO	1500

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H

Manufacturer	Brand	Model
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G (OEM Only)
AMD	Ryzen™ 7	4700GE (OEM Only)
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 9	3900 Processor (OEM Only)
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9 PRO	3900

Manufacturer	Brand	Model
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 20H2 Supported Intel Processors

Article • 02/26/2024

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Atom®	x6200FE
Intel®	Atom®	x6211E
Intel®	Atom®	x6212RE
Intel®	Atom®	x6413E
Intel®	Atom®	x6414RE
Intel®	Atom®	x6425E
Intel®	Atom®	x6425RE
Intel®	Atom®	x6427FE
Intel®	Celeron®	6305
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3865U
Intel®	Celeron®	3867U
Intel®	Celeron®	3955U
Intel®	Celeron®	3965U
Intel®	Celeron®	3965Y

Manufacturer	Brand	Model
Intel®	Celeron®	4205U
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	6305E
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Celeron®	G3930
Intel®	Celeron®	G3930E
Intel®	Celeron®	G3930T
Intel®	Celeron®	G3930TE
Intel®	Celeron®	G3950
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900

Manufacturer	Brand	Model
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925
Intel®	Celeron®	J4005
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105
Intel®	Celeron®	J4125
Intel®	Celeron®	J6412
Intel®	Celeron®	J6413
Intel®	Celeron®	N4000
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100
Intel®	Celeron®	N4120
Intel®	Celeron®	N4500
Intel®	Celeron®	N4505
Intel®	Celeron®	N5100
Intel®	Celeron®	N5105
Intel®	Celeron®	N6210
Intel®	Celeron®	N6211
Intel®	Core™	3-100U
Intel®	Core™	3-100U with IPU
Intel®	Core™	5-120U

Manufacturer	Brand	Model
Intel®	Core™	7-150U
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4
Intel®	Core™	i3-1005G1
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-1110G4
Intel®	Core™	i3-1115G4
Intel®	Core™	i3-1115G4E
Intel®	Core™	i3-1115GRE
Intel®	Core™	i3-1120G4

Manufacturer	Brand	Model
Intel®	Core™	i3-1125G4
Intel®	Core™	i3-14100
Intel®	Core™	i3-14100F
Intel®	Core™	i3-14100T
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i3-7020U
Intel®	Core™	i3-7100
Intel®	Core™	i3-7100E

Manufacturer	Brand	Model
Intel®	Core™	i3-7100H
Intel®	Core™	i3-7100T
Intel®	Core™	i3-7100U
Intel®	Core™	i3-7101E
Intel®	Core™	i3-7101TE
Intel®	Core™	i3-7102E
Intel®	Core™	i3-7130U
Intel®	Core™	i3-7167U
Intel®	Core™	i3-7300
Intel®	Core™	i3-7300T
Intel®	Core™	i3-7320
Intel®	Core™	i3-7350K
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E

Manufacturer	Brand	Model
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i3-L13G4
Intel®	Core™	i5+8400
Intel®	Core™	i5+8500
Intel®	Core™	i5-10200H
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F

Manufacturer	Brand	Model
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T
Intel®	Core™	i5-10500TE
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T
Intel®	Core™	i5-11300H
Intel®	Core™	i5-1130G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-11400
Intel®	Core™	i5-11400F
Intel®	Core™	i5-11400T
Intel®	Core™	i5-1140G7
Intel®	Core™	i5-1145G7
Intel®	Core™	i5-1145G7E
Intel®	Core™	i5-1145GRE
Intel®	Core™	i5-11500
Intel®	Core™	i5-11500T
Intel®	Core™	i5-11600
Intel®	Core™	i5-11600K

Manufacturer	Brand	Model
Intel®	Core™	i5-11600KF
Intel®	Core™	i5-11600T
Intel®	Core™	i5-14400
Intel®	Core™	i5-14400F
Intel®	Core™	i5-14400T
Intel®	Core™	i5-14500
Intel®	Core™	i5-14500T
Intel®	Core™	i5-14600
Intel®	Core™	i5-14600K
Intel®	Core™	i5-14600KF
Intel®	Core™	i5-14600T
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U
Intel®	Core™	i5-6300HQ

Manufacturer	Brand	Model
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i5-7200U
Intel®	Core™	i5-7260U
Intel®	Core™	i5-7267U
Intel®	Core™	i5-7287U
Intel®	Core™	i5-7300HQ
Intel®	Core™	i5-7300U
Intel®	Core™	i5-7360U
Intel®	Core™	i5-7400
Intel®	Core™	i5-7400T

Manufacturer	Brand	Model
Intel®	Core™	i5-7440EQ
Intel®	Core™	i5-7440HQ
Intel®	Core™	i5-7442EQ
Intel®	Core™	i5-7500
Intel®	Core™	i5-7500T
Intel®	Core™	i5-7600
Intel®	Core™	i5-7600K
Intel®	Core™	i5-7600T
Intel®	Core™	i5-7640X
Intel®	Core™	i5-7Y54
Intel®	Core™	i5-7Y57
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400

Manufacturer	Brand	Model
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i5-L16G7
Intel®	Core™	i7+8700

Manufacturer	Brand	Model
Intel®	Core™	i7-10510U
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-11370H
Intel®	Core™	i7-11375H
Intel®	Core™	i7-1160G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-11700
Intel®	Core™	i7-11700F

Manufacturer	Brand	Model
Intel®	Core™	i7-11700K
Intel®	Core™	i7-11700KF
Intel®	Core™	i7-11700T
Intel®	Core™	i7-1180G7
Intel®	Core™	i7-1185G7
Intel®	Core™	i7-1185G7E
Intel®	Core™	i7-1185GRE
Intel®	Core™	i7-14700
Intel®	Core™	i7-14700F
Intel®	Core™	i7-14700K
Intel®	Core™	i7-14700KF
Intel®	Core™	i7-14700T
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U

Manufacturer	Brand	Model
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7500U
Intel®	Core™	i7-7560U
Intel®	Core™	i7-7567U
Intel®	Core™	i7-7600U
Intel®	Core™	i7-7660U
Intel®	Core™	i7-7700
Intel®	Core™	i7-7700HQ

Manufacturer	Brand	Model
Intel®	Core™	i7-7700K
Intel®	Core™	i7-7700T
Intel®	Core™	i7-7740X
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820EQ
Intel®	Core™	i7-7820HK
Intel®	Core™	i7-7820HQ
Intel®	Core™	i7-7820X
Intel®	Core™	i7-7920HQ
Intel®	Core™	i7-7Y75
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8850H

Manufacturer	Brand	Model
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK

Manufacturer	Brand	Model
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-11900
Intel®	Core™	i9-11900F
Intel®	Core™	i9-11900K
Intel®	Core™	i9-11900KF
Intel®	Core™	i9-11900T
Intel®	Core™	i9-14900
Intel®	Core™	i9-14900F
Intel®	Core™	i9-14900K
Intel®	Core™	i9-14900KF
Intel®	Core™	i9-14900T
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X

Manufacturer	Brand	Model
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m3-7Y30
Intel®	Core™	m3-7Y32
Intel®	Core™	m3-8100Y
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Core™	Ultra 5-125H
Intel®	Core™	Ultra 5-125U
Intel®	Core™	Ultra 5-134U
Intel®	Core™	Ultra 5-135H
Intel®	Core™	Ultra 5-135U
Intel®	Core™	Ultra 7-155H
Intel®	Core™	Ultra 7-155U
Intel®	Core™	Ultra 7-164U

Manufacturer	Brand	Model
Intel®	Core™	Ultra 7-165H
Intel®	Core™	Ultra 7-165U
Intel®	Core™	Ultra 9-185H
Intel®	Pentium®	6805
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Pentium®	G4560
Intel®	Pentium®	G4560T
Intel®	Pentium®	G4600
Intel®	Pentium®	G4600T
Intel®	Pentium®	G4620
Intel®	Pentium®	Gold 4410Y
Intel®	Pentium®	Gold 4415U

Manufacturer	Brand	Model
Intel®	Pentium®	Gold 4415Y
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold 7505
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	J6426
Intel®	Pentium®	N6415
Intel®	Pentium®	Silver J5005
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000
Intel®	Pentium®	Silver N5030
Intel®	Pentium®	Silver N6000
Intel®	Pentium®	Silver N6005
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537
Intel®	Xeon®	D-1539

Manufacturer	Brand	Model
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G

Manufacturer	Brand	Model
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE

Manufacturer	Brand	Model
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1220 v6
Intel®	Xeon®	E3-1225 v5
Intel®	Xeon®	E3-1225 v6
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1230 v6
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240 v6
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1245 v6
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1270 v6
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1275 v6
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1280 v6
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285 v6
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1501L v6
Intel®	Xeon®	E3-1501M v6
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505L v6
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1505M v6
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1535M v6
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L
Intel®	Xeon®	Gold 6240R
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold Gold 5218
Intel®	Xeon®	Gold Gold 5318Y
Intel®	Xeon®	Gold Gold 6238R
Intel®	Xeon®	Gold6246
Intel®	Xeon®	Goldv 6234
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-2223

Manufacturer	Brand	Model
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M

Windows 10 20H2 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	850
Qualcomm®	Snapdragon™	8cx

Windows 10 21H1 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G (OEM Only)
AMD	Athlon™	Gold 3150GE (OEM Only)
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE (OEM Only)
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940

Manufacturer	Brand	Model
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7232P
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	72F3
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7302P
AMD	EPYC™	7313
AMD	EPYC™	7313P
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7351P
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	73F3
AMD	EPYC™	7401
AMD	EPYC™	7401P
AMD	EPYC™	7402
AMD	EPYC™	7402P

Manufacturer	Brand	Model
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7443P
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	74F3
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7502P
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7543P
AMD	EPYC™	7551
AMD	EPYC™	7551P
AMD	EPYC™	7552
AMD	EPYC™	75F3
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7702P

Manufacturer	Brand	Model
AMD	EPYC™	7713
AMD	EPYC™	7713P
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300

Manufacturer	Brand	Model
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics

Manufacturer	Brand	Model
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3100

Manufacturer	Brand	Model
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE (OEM Only)
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3300X
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G (OEM Only)
AMD	Ryzen™ 3	4300GE (OEM Only)
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	1600 (AF)

Manufacturer	Brand	Model
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE (OEM Only)
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor (OEM Only)
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G (OEM Only)
AMD	Ryzen™ 5	4600GE (OEM Only)
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5 PRO	1500

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 7	1700 Processor
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H

Manufacturer	Brand	Model
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G (OEM Only)
AMD	Ryzen™ 7	4700GE (OEM Only)
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	1700X Processor
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 9	3900 Processor (OEM Only)
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9 PRO	3900

Manufacturer	Brand	Model
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 21H1 Supported Intel Processors

Article • 05/18/2022

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Intel®	Atom®	E3805	IoT Enterprise Only
Intel®	Atom®	E3815	IoT Enterprise Only
Intel®	Atom®	E3825	IoT Enterprise Only
Intel®	Atom®	E3826	IoT Enterprise Only
Intel®	Atom®	E3827	IoT Enterprise Only
Intel®	Atom®	E3845	IoT Enterprise Only
Intel®	Atom®	x5-E3930	IoT Enterprise Only
Intel®	Atom®	x5-E3940	IoT Enterprise Only
Intel®	Atom®	x5-E3950	IoT Enterprise Only
Intel®	Atom®	x5-E8000	IoT Enterprise Only
Intel®	Atom®	x6200FE	
Intel®	Atom®	x6211E	
Intel®	Atom®	x6212RE	
Intel®	Atom®	x6413E	
Intel®	Atom®	x6414RE	
Intel®	Atom®	x6425E	
Intel®	Atom®	x6425RE	
Intel®	Atom®	x6427FE	
Intel®	Celeron®	6305	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	3205U	
Intel®	Celeron®	3215U	
Intel®	Celeron®	3755U	
Intel®	Celeron®	3765U	
Intel®	Celeron®	3855U	
Intel®	Celeron®	3865U	
Intel®	Celeron®	3867U	
Intel®	Celeron®	3955U	
Intel®	Celeron®	3965U	
Intel®	Celeron®	3965Y	
Intel®	Celeron®	4205U	
Intel®	Celeron®	4305U	
Intel®	Celeron®	4305UE	
Intel®	Celeron®	5205U	
Intel®	Celeron®	5305U	
Intel®	Celeron®	6305E	
Intel®	Celeron®	G3900	
Intel®	Celeron®	G3900E	
Intel®	Celeron®	G3900T	
Intel®	Celeron®	G3900TE	
Intel®	Celeron®	G3902E	
Intel®	Celeron®	G3920	
Intel®	Celeron®	G3930	
Intel®	Celeron®	G3930E	
Intel®	Celeron®	G3930T	
Intel®	Celeron®	G3930TE	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	G3950	
Intel®	Celeron®	G4900	
Intel®	Celeron®	G4900T	
Intel®	Celeron®	G4920	
Intel®	Celeron®	G4930	
Intel®	Celeron®	G4930E	
Intel®	Celeron®	G4930T	
Intel®	Celeron®	G4932E	
Intel®	Celeron®	G4950	
Intel®	Celeron®	G5900	
Intel®	Celeron®	G5900E	
Intel®	Celeron®	G5900T	
Intel®	Celeron®	G5900TE	
Intel®	Celeron®	G5905	
Intel®	Celeron®	G5905T	
Intel®	Celeron®	G5920	
Intel®	Celeron®	G5925	
Intel®	Celeron®	J1900	IoT Enterprise Only
Intel®	Celeron®	J3355	IoT Enterprise Only
Intel®	Celeron®	J3355E	IoT Enterprise Only
Intel®	Celeron®	J3455	IoT Enterprise Only
Intel®	Celeron®	J3455E	IoT Enterprise Only
Intel®	Celeron®	J4005	
Intel®	Celeron®	J4025	
Intel®	Celeron®	J4105	
Intel®	Celeron®	J4125	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	J6412	
Intel®	Celeron®	J6413	
Intel®	Celeron®	N2807	IoT Enterprise Only
Intel®	Celeron®	N2920	IoT Enterprise Only
Intel®	Celeron®	N2930	IoT Enterprise Only
Intel®	Celeron®	N3010	IoT Enterprise Only
Intel®	Celeron®	N3060	IoT Enterprise Only
Intel®	Celeron®	N3160	IoT Enterprise Only
Intel®	Celeron®	N3350	IoT Enterprise Only
Intel®	Celeron®	N4000	
Intel®	Celeron®	N4020	
Intel®	Celeron®	N4100	
Intel®	Celeron®	N4120	
Intel®	Celeron®	N4500	
Intel®	Celeron®	N4505	
Intel®	Celeron®	N5100	
Intel®	Celeron®	N5105	
Intel®	Celeron®	N6210	
Intel®	Celeron®	N6211	
Intel®	Core™	3-100U	
Intel®	Core™	3-100U with IPU	
Intel®	Core™	5-120U	
Intel®	Core™	7-150U	
Intel®	Core™	i3-1000G1	
Intel®	Core™	i3-1000G4	
Intel®	Core™	i3-1005G1	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-10100	
Intel®	Core™	i3-10100E	
Intel®	Core™	i3-10100F	
Intel®	Core™	i3-10100T	
Intel®	Core™	i3-10100TE	
Intel®	Core™	i3-10100Y	
Intel®	Core™	i3-10105	
Intel®	Core™	i3-10105F	
Intel®	Core™	i3-10105T	
Intel®	Core™	i3-10110U	
Intel®	Core™	i3-10110Y	
Intel®	Core™	i3-10300	
Intel®	Core™	i3-10300T	
Intel®	Core™	i3-10305	
Intel®	Core™	i3-10305T	
Intel®	Core™	i3-10320	
Intel®	Core™	i3-10325	
Intel®	Core™	i3-1110G4	
Intel®	Core™	i3-1115G4	
Intel®	Core™	i3-1115G4E	
Intel®	Core™	i3-1115GRE	
Intel®	Core™	i3-1120G4	
Intel®	Core™	i3-1125G4	
Intel®	Core™	i3-14100	
Intel®	Core™	i3-14100F	
Intel®	Core™	i3-14100T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-5005U	
Intel®	Core™	i3-5010U	
Intel®	Core™	i3-5015U	
Intel®	Core™	i3-5020U	
Intel®	Core™	i3-5157U	
Intel®	Core™	i3-6006U	
Intel®	Core™	i3-6098P	
Intel®	Core™	i3-6100	
Intel®	Core™	i3-6100E	
Intel®	Core™	i3-6100H	
Intel®	Core™	i3-6100T	
Intel®	Core™	i3-6100TE	
Intel®	Core™	i3-6100U	
Intel®	Core™	i3-6102E	
Intel®	Core™	i3-6157U	
Intel®	Core™	i3-6167U	
Intel®	Core™	i3-6300	
Intel®	Core™	i3-6300T	
Intel®	Core™	i3-6320	
Intel®	Core™	i3-7020U	
Intel®	Core™	i3-7100	
Intel®	Core™	i3-7100E	
Intel®	Core™	i3-7100H	
Intel®	Core™	i3-7100T	
Intel®	Core™	i3-7100U	
Intel®	Core™	i3-7101E	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-7101TE	
Intel®	Core™	i3-7102E	
Intel®	Core™	i3-7130U	
Intel®	Core™	i3-7167U	
Intel®	Core™	i3-7300	
Intel®	Core™	i3-7300T	
Intel®	Core™	i3-7320	
Intel®	Core™	i3-7350K	
Intel®	Core™	i3-8100	
Intel®	Core™	i3-8100B	
Intel®	Core™	i3-8100H	
Intel®	Core™	i3-8100T	
Intel®	Core™	i3-8109U	
Intel®	Core™	i3-8130U	
Intel®	Core™	i3-8140U	
Intel®	Core™	i3-8145U	
Intel®	Core™	i3-8145UE	
Intel®	Core™	i3-8300	
Intel®	Core™	i3-8300T	
Intel®	Core™	i3-8350K	
Intel®	Core™	i3-9100	
Intel®	Core™	i3-9100E	
Intel®	Core™	i3-9100F	
Intel®	Core™	i3-9100HL	
Intel®	Core™	i3-9100T	
Intel®	Core™	i3-9100TE	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-9300	
Intel®	Core™	i3-9300T	
Intel®	Core™	i3-9320	
Intel®	Core™	i3-9350K	
Intel®	Core™	i3-9350KF	
Intel®	Core™	i3-L13G4	
Intel®	Core™	i5+8400	
Intel®	Core™	i5+8500	
Intel®	Core™	i5-10200H	
Intel®	Core™	i5-10210U	
Intel®	Core™	i5-10210Y	
Intel®	Core™	i5-10300H	
Intel®	Core™	i5-1030G4	
Intel®	Core™	i5-1030G7	
Intel®	Core™	i5-10310U	
Intel®	Core™	i5-10310Y	
Intel®	Core™	i5-1035G1	
Intel®	Core™	i5-1035G4	
Intel®	Core™	i5-1035G7	
Intel®	Core™	i5-1038NG7	
Intel®	Core™	i5-10400	
Intel®	Core™	i5-10400F	
Intel®	Core™	i5-10400H	
Intel®	Core™	i5-10400T	
Intel®	Core™	i5-10500	
Intel®	Core™	i5-10500E	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-10500H	
Intel®	Core™	i5-10500T	
Intel®	Core™	i5-10500TE	
Intel®	Core™	i5-10600	
Intel®	Core™	i5-10600K	
Intel®	Core™	i5-10600KF	
Intel®	Core™	i5-10600T	
Intel®	Core™	i5-11300H	
Intel®	Core™	i5-1130G7	
Intel®	Core™	i5-1135G7	
Intel®	Core™	i5-11400	
Intel®	Core™	i5-11400F	
Intel®	Core™	i5-11400T	
Intel®	Core™	i5-1140G7	
Intel®	Core™	i5-1145G7	
Intel®	Core™	i5-1145G7E	
Intel®	Core™	i5-1145GRE	
Intel®	Core™	i5-11500	
Intel®	Core™	i5-11500T	
Intel®	Core™	i5-11600	
Intel®	Core™	i5-11600K	
Intel®	Core™	i5-11600KF	
Intel®	Core™	i5-11600T	
Intel®	Core™	i5-14400	
Intel®	Core™	i5-14400F	
Intel®	Core™	i5-14400T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-14500	
Intel®	Core™	i5-14500T	
Intel®	Core™	i5-14600	
Intel®	Core™	i5-14600K	
Intel®	Core™	i5-14600KF	
Intel®	Core™	i5-14600T	
Intel®	Core™	i5-5200U	
Intel®	Core™	i5-5250U	
Intel®	Core™	i5-5257U	
Intel®	Core™	i5-5287U	
Intel®	Core™	i5-5300U	
Intel®	Core™	i5-5350H	
Intel®	Core™	i5-5350U	
Intel®	Core™	i5-5575R	
Intel®	Core™	i5-5675C	
Intel®	Core™	i5-5675R	
Intel®	Core™	i5-6200U	
Intel®	Core™	i5-6260U	
Intel®	Core™	i5-6267U	
Intel®	Core™	i5-6287U	
Intel®	Core™	i5-6300HQ	
Intel®	Core™	i5-6300U	
Intel®	Core™	i5-6350HQ	
Intel®	Core™	i5-6360U	
Intel®	Core™	i5-6400	
Intel®	Core™	i5-6400T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-6402P	
Intel®	Core™	i5-6440EQ	
Intel®	Core™	i5-6440HQ	
Intel®	Core™	i5-6442EQ	
Intel®	Core™	i5-6500	
Intel®	Core™	i5-6500T	
Intel®	Core™	i5-6500TE	
Intel®	Core™	i5-6585R	
Intel®	Core™	i5-6600	
Intel®	Core™	i5-6600K	
Intel®	Core™	i5-6600T	
Intel®	Core™	i5-6685R	
Intel®	Core™	i5-7200U	
Intel®	Core™	i5-7260U	
Intel®	Core™	i5-7267U	
Intel®	Core™	i5-7287U	
Intel®	Core™	i5-7300HQ	
Intel®	Core™	i5-7300U	
Intel®	Core™	i5-7360U	
Intel®	Core™	i5-7400	
Intel®	Core™	i5-7400T	
Intel®	Core™	i5-7440EQ	
Intel®	Core™	i5-7440HQ	
Intel®	Core™	i5-7442EQ	
Intel®	Core™	i5-7500	
Intel®	Core™	i5-7500T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-7600	
Intel®	Core™	i5-7600K	
Intel®	Core™	i5-7600T	
Intel®	Core™	i5-7640X	
Intel®	Core™	i5-7Y54	
Intel®	Core™	i5-7Y57	
Intel®	Core™	i5-8200Y	
Intel®	Core™	i5-8210Y	
Intel®	Core™	i5-8250U	
Intel®	Core™	i5-8257U	
Intel®	Core™	i5-8259U	
Intel®	Core™	i5-8260U	
Intel®	Core™	i5-8265U	
Intel®	Core™	i5-8269U	
Intel®	Core™	i5-8279U	
Intel®	Core™	i5-8300H	
Intel®	Core™	i5-8310Y	
Intel®	Core™	i5-8350U	
Intel®	Core™	i5-8365U	
Intel®	Core™	i5-8365UE	
Intel®	Core™	i5-8400	
Intel®	Core™	i5-8400B	
Intel®	Core™	i5-8400H	
Intel®	Core™	i5-8400T	
Intel®	Core™	i5-8500	
Intel®	Core™	i5-8500B	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-8500T	
Intel®	Core™	i5-8600	
Intel®	Core™	i5-8600K	
Intel®	Core™	i5-8600T	
Intel®	Core™	i5-9300H	
Intel®	Core™	i5-9300HF	
Intel®	Core™	i5-9400	
Intel®	Core™	i5-9400F	
Intel®	Core™	i5-9400H	
Intel®	Core™	i5-9400T	
Intel®	Core™	i5-9500	
Intel®	Core™	i5-9500E	
Intel®	Core™	i5-9500F	
Intel®	Core™	i5-9500T	
Intel®	Core™	i5-9500TE	
Intel®	Core™	i5-9600	
Intel®	Core™	i5-9600K	
Intel®	Core™	i5-9600KF	
Intel®	Core™	i5-9600T	
Intel®	Core™	i5-L16G7	
Intel®	Core™	i7+8700	
Intel®	Core™	i7-10510U	
Intel®	Core™	i7-10510Y	
Intel®	Core™	i7-1060G7	
Intel®	Core™	i7-10610U	
Intel®	Core™	i7-1065G7	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-1068NG7	
Intel®	Core™	i7-10700	
Intel®	Core™	i7-10700E	
Intel®	Core™	i7-10700F	
Intel®	Core™	i7-10700K	
Intel®	Core™	i7-10700KF	
Intel®	Core™	i7-10700T	
Intel®	Core™	i7-10700TE	
Intel®	Core™	i7-10710U	
Intel®	Core™	i7-10750H	
Intel®	Core™	i7-10810U	
Intel®	Core™	i7-10850H	
Intel®	Core™	i7-10870H	
Intel®	Core™	i7-10875H	
Intel®	Core™	i7-11370H	
Intel®	Core™	i7-11375H	
Intel®	Core™	i7-1160G7	
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-11700	
Intel®	Core™	i7-11700F	
Intel®	Core™	i7-11700K	
Intel®	Core™	i7-11700KF	
Intel®	Core™	i7-11700T	
Intel®	Core™	i7-1180G7	
Intel®	Core™	i7-1185G7	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-1185G7E	
Intel®	Core™	i7-1185GRE	
Intel®	Core™	i7-14700	
Intel®	Core™	i7-14700F	
Intel®	Core™	i7-14700K	
Intel®	Core™	i7-14700KF	
Intel®	Core™	i7-14700T	
Intel®	Core™	i7-5500U	
Intel®	Core™	i7-5550U	
Intel®	Core™	i7-5557U	
Intel®	Core™	i7-5600U	
Intel®	Core™	i7-5650U	
Intel®	Core™	i7-5700EQ	
Intel®	Core™	i7-5700HQ	
Intel®	Core™	i7-5750HQ	
Intel®	Core™	i7-5775C	
Intel®	Core™	i7-5775R	
Intel®	Core™	i7-5850EQ	
Intel®	Core™	i7-5850HQ	
Intel®	Core™	i7-5950HQ	
Intel®	Core™	i7-6500U	
Intel®	Core™	i7-6560U	
Intel®	Core™	i7-6567U	
Intel®	Core™	i7-6600U	
Intel®	Core™	i7-6650U	
Intel®	Core™	i7-6660U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-6700	
Intel®	Core™	i7-6700HQ	
Intel®	Core™	i7-6700K	
Intel®	Core™	i7-6700T	
Intel®	Core™	i7-6700TE	
Intel®	Core™	i7-6770HQ	
Intel®	Core™	i7-6785R	
Intel®	Core™	i7-6820EQ	
Intel®	Core™	i7-6820HK	
Intel®	Core™	i7-6820HQ	
Intel®	Core™	i7-6822EQ	
Intel®	Core™	i7-6870HQ	
Intel®	Core™	i7-6920HQ	
Intel®	Core™	i7-6970HQ	
Intel®	Core™	i7-7500U	
Intel®	Core™	i7-7560U	
Intel®	Core™	i7-7567U	
Intel®	Core™	i7-7600U	
Intel®	Core™	i7-7660U	
Intel®	Core™	i7-7700	
Intel®	Core™	i7-7700HQ	
Intel®	Core™	i7-7700K	
Intel®	Core™	i7-7700T	
Intel®	Core™	i7-7740X	
Intel®	Core™	i7-7800X	
Intel®	Core™	i7-7820EQ	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-7820HK	
Intel®	Core™	i7-7820HQ	
Intel®	Core™	i7-7820X	
Intel®	Core™	i7-7920HQ	
Intel®	Core™	i7-7Y75	
Intel®	Core™	i7-8086K	
Intel®	Core™	i7-8500Y	
Intel®	Core™	i7-8550U	
Intel®	Core™	i7-8557U	
Intel®	Core™	i7-8559U	
Intel®	Core™	i7-8565U	
Intel®	Core™	i7-8569U	
Intel®	Core™	i7-8650U	
Intel®	Core™	i7-8665U	
Intel®	Core™	i7-8665UE	
Intel®	Core™	i7-8700	
Intel®	Core™	i7-8700B	
Intel®	Core™	i7-8700K	
Intel®	Core™	i7-8700T	
Intel®	Core™	i7-8750H	
Intel®	Core™	i7-8850H	
Intel®	Core™	i7-9700	
Intel®	Core™	i7-9700E	
Intel®	Core™	i7-9700F	
Intel®	Core™	i7-9700K	
Intel®	Core™	i7-9700KF	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-9700T	
Intel®	Core™	i7-9700TE	
Intel®	Core™	i7-9750H	
Intel®	Core™	i7-9750HF	
Intel®	Core™	i7-9800X	
Intel®	Core™	i7-9850H	
Intel®	Core™	i7-9850HE	
Intel®	Core™	i7-9850HL	
Intel®	Core™	i9-10850K	
Intel®	Core™	i9-10885H	
Intel®	Core™	i9-10900	
Intel®	Core™	i9-10900E	
Intel®	Core™	i9-10900F	
Intel®	Core™	i9-10900K	
Intel®	Core™	i9-10900KF	
Intel®	Core™	i9-10900T	
Intel®	Core™	i9-10900TE	
Intel®	Core™	i9-10900X	
Intel®	Core™	i9-10920X	
Intel®	Core™	i9-10940X	
Intel®	Core™	i9-10980HK	
Intel®	Core™	i9-10980XE	
Intel®	Core™	i9-11900	
Intel®	Core™	i9-11900F	
Intel®	Core™	i9-11900K	
Intel®	Core™	i9-11900KF	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-11900T	
Intel®	Core™	i9-14900	
Intel®	Core™	i9-14900F	
Intel®	Core™	i9-14900K	
Intel®	Core™	i9-14900KF	
Intel®	Core™	i9-14900T	
Intel®	Core™	i9-7900X	
Intel®	Core™	i9-7920X	
Intel®	Core™	i9-7940X	
Intel®	Core™	i9-7960X	
Intel®	Core™	i9-7980XE	
Intel®	Core™	i9-8950HK	
Intel®	Core™	i9-9820X	
Intel®	Core™	i9-9880H	
Intel®	Core™	i9-9900	
Intel®	Core™	i9-9900K	
Intel®	Core™	i9-9900KF	
Intel®	Core™	i9-9900KS	
Intel®	Core™	i9-9900T	
Intel®	Core™	i9-9900X	
Intel®	Core™	i9-9920X	
Intel®	Core™	i9-9940X	
Intel®	Core™	i9-9960X	
Intel®	Core™	i9-9980HK	
Intel®	Core™	i9-9980XE	
Intel®	Core™	m3-6Y30	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	m3-7Y30	
Intel®	Core™	m3-7Y32	
Intel®	Core™	m3-8100Y	
Intel®	Core™	m5-6Y54	
Intel®	Core™	m5-6Y57	
Intel®	Core™	M-5Y10	
Intel®	Core™	M-5Y10a	
Intel®	Core™	M-5Y10c	
Intel®	Core™	M-5Y31	
Intel®	Core™	M-5Y51	
Intel®	Core™	M-5Y70	
Intel®	Core™	M-5Y71	
Intel®	Core™	m7-6Y75	
Intel®	Core™	Ultra 5-125H	
Intel®	Core™	Ultra 5-125U	
Intel®	Core™	Ultra 5-134U	
Intel®	Core™	Ultra 5-135H	
Intel®	Core™	Ultra 5-135U	
Intel®	Core™	Ultra 7-155H	
Intel®	Core™	Ultra 7-155U	
Intel®	Core™	Ultra 7-164U	
Intel®	Core™	Ultra 7-165H	
Intel®	Core™	Ultra 7-165U	
Intel®	Core™	Ultra 9-185H	
Intel®	Pentium®	6805	
Intel®	Pentium®	3805U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	3825U	
Intel®	Pentium®	4405U	
Intel®	Pentium®	4405Y	
Intel®	Pentium®	D1507	
Intel®	Pentium®	D1508	
Intel®	Pentium®	D1509	
Intel®	Pentium®	D1517	
Intel®	Pentium®	D1519	
Intel®	Pentium®	G4400	
Intel®	Pentium®	G4400T	
Intel®	Pentium®	G4400TE	
Intel®	Pentium®	G4500	
Intel®	Pentium®	G4500T	
Intel®	Pentium®	G4520	
Intel®	Pentium®	G4560	
Intel®	Pentium®	G4560T	
Intel®	Pentium®	G4600	
Intel®	Pentium®	G4600T	
Intel®	Pentium®	G4620	
Intel®	Pentium®	Gold 4410Y	
Intel®	Pentium®	Gold 4415U	
Intel®	Pentium®	Gold 4415Y	
Intel®	Pentium®	Gold 4417U	
Intel®	Pentium®	Gold 4425Y	
Intel®	Pentium®	Gold 5405U	
Intel®	Pentium®	Gold 6405U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	Gold 6500Y	
Intel®	Pentium®	Gold 7505	
Intel®	Pentium®	Gold G5400	
Intel®	Pentium®	Gold G5400T	
Intel®	Pentium®	Gold G5420	
Intel®	Pentium®	Gold G5420T	
Intel®	Pentium®	Gold G5500	
Intel®	Pentium®	Gold G5500T	
Intel®	Pentium®	Gold G5600	
Intel®	Pentium®	Gold G5600T	
Intel®	Pentium®	Gold G5620	
Intel®	Pentium®	Gold G6400	
Intel®	Pentium®	Gold G6400E	
Intel®	Pentium®	Gold G6400T	
Intel®	Pentium®	Gold G6400TE	
Intel®	Pentium®	Gold G6405	
Intel®	Pentium®	Gold G6405T	
Intel®	Pentium®	Gold G6500	
Intel®	Pentium®	Gold G6500T	
Intel®	Pentium®	Gold G6505	
Intel®	Pentium®	Gold G6505T	
Intel®	Pentium®	Gold G6600	
Intel®	Pentium®	Gold G6605	
Intel®	Pentium®	J6426	
Intel®	Pentium®	N3710	IoT Enterprise Only
Intel®	Pentium®	N4200	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	N6415	
Intel®	Pentium®	Silver J5005	
Intel®	Pentium®	Silver J5040	
Intel®	Pentium®	Silver N5000	
Intel®	Pentium®	Silver N5030	
Intel®	Pentium®	Silver N6000	
Intel®	Pentium®	Silver N6005	
Intel®	Xeon®	Bronze 3104	
Intel®	Xeon®	Bronze 3106	
Intel®	Xeon®	Bronze 3204	
Intel®	Xeon®	Bronze 3206R	
Intel®	Xeon®	D-1513N	
Intel®	Xeon®	D-1518	
Intel®	Xeon®	D-1520	
Intel®	Xeon®	D-1521	
Intel®	Xeon®	D-1523N	
Intel®	Xeon®	D-1527	
Intel®	Xeon®	D-1528	
Intel®	Xeon®	D-1529	
Intel®	Xeon®	D-1531	
Intel®	Xeon®	D-1533N	
Intel®	Xeon®	D-1537	
Intel®	Xeon®	D-1539	
Intel®	Xeon®	D-1540	
Intel®	Xeon®	D-1541	
Intel®	Xeon®	D-1543N	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	D-1548	
Intel®	Xeon®	D-1553N	
Intel®	Xeon®	D-1557	
Intel®	Xeon®	D-1559	
Intel®	Xeon®	D-1567	
Intel®	Xeon®	D-1571	
Intel®	Xeon®	D-1577	
Intel®	Xeon®	D-2123IT	
Intel®	Xeon®	D-2141I	
Intel®	Xeon®	D-2142IT	
Intel®	Xeon®	D-2143IT	
Intel®	Xeon®	D-2145NT	
Intel®	Xeon®	D-2146NT	
Intel®	Xeon®	D-2161I	
Intel®	Xeon®	D-2163IT	
Intel®	Xeon®	D-2166NT	
Intel®	Xeon®	D-2173IT	
Intel®	Xeon®	D-2177NT	
Intel®	Xeon®	D-2183IT	
Intel®	Xeon®	D-2187NT	
Intel®	Xeon®	E-2124	
Intel®	Xeon®	E-2124G	
Intel®	Xeon®	E-2126G	
Intel®	Xeon®	E-2134	
Intel®	Xeon®	E-2136	
Intel®	Xeon®	E-2144G	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2146G	
Intel®	Xeon®	E-2174G	
Intel®	Xeon®	E-2176G	
Intel®	Xeon®	E-2176M	
Intel®	Xeon®	E-2186G	
Intel®	Xeon®	E-2186M	
Intel®	Xeon®	E-2224	
Intel®	Xeon®	E-2224G	
Intel®	Xeon®	E-2226G	
Intel®	Xeon®	E-2226GE	
Intel®	Xeon®	E-2234	
Intel®	Xeon®	E-2236	
Intel®	Xeon®	E-2244G	
Intel®	Xeon®	E-2246G	
Intel®	Xeon®	E-2254ME	
Intel®	Xeon®	E-2254ML	
Intel®	Xeon®	E-2274G	
Intel®	Xeon®	E-2276G	
Intel®	Xeon®	E-2276M	
Intel®	Xeon®	E-2276ME	
Intel®	Xeon®	E-2276ML	
Intel®	Xeon®	E-2278G	
Intel®	Xeon®	E-2278GE	
Intel®	Xeon®	E-2278GEL	
Intel®	Xeon®	E-2286G	
Intel®	Xeon®	E-2286M	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2288G	
Intel®	Xeon®	E3-1220 v5	
Intel®	Xeon®	E3-1220 v6	
Intel®	Xeon®	E3-1225 v5	
Intel®	Xeon®	E3-1225 v6	
Intel®	Xeon®	E3-1230 v5	
Intel®	Xeon®	E3-1230 v6	
Intel®	Xeon®	E3-1235L v5	
Intel®	Xeon®	E3-1240 v5	
Intel®	Xeon®	E3-1240 v6	
Intel®	Xeon®	E3-1240L v5	
Intel®	Xeon®	E3-1245 v5	
Intel®	Xeon®	E3-1245 v6	
Intel®	Xeon®	E3-1258L v4	
Intel®	Xeon®	E3-1260L v5	
Intel®	Xeon®	E3-1265L v4	
Intel®	Xeon®	E3-1268L v5	
Intel®	Xeon®	E3-1270 v5	
Intel®	Xeon®	E3-1270 v6	
Intel®	Xeon®	E3-1275 v5	
Intel®	Xeon®	E3-1275 v6	
Intel®	Xeon®	E3-1278L v4	
Intel®	Xeon®	E3-1280 v5	
Intel®	Xeon®	E3-1280 v6	
Intel®	Xeon®	E3-1285 v4	
Intel®	Xeon®	E3-1285 v6	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1285L v4	
Intel®	Xeon®	E3-1501L v6	
Intel®	Xeon®	E3-1501M v6	
Intel®	Xeon®	E3-1505L v5	
Intel®	Xeon®	E3-1505L v6	
Intel®	Xeon®	E3-1505M v5	
Intel®	Xeon®	E3-1505M v6	
Intel®	Xeon®	E3-1515M v5	
Intel®	Xeon®	E3-1535M v5	
Intel®	Xeon®	E3-1535M v6	
Intel®	Xeon®	E3-1545M v5	
Intel®	Xeon®	E3-1558L v5	
Intel®	Xeon®	E3-1565L v5	
Intel®	Xeon®	E3-1575M v5	
Intel®	Xeon®	E3-1578L v5	
Intel®	Xeon®	E3-1585 v5	
Intel®	Xeon®	E3-1585L v5	
Intel®	Xeon®	E5-1620 v4	
Intel®	Xeon®	E5-1630 v4	
Intel®	Xeon®	E5-1650 v4	
Intel®	Xeon®	E5-1660 v4	
Intel®	Xeon®	E5-1680 v4	
Intel®	Xeon®	E5-2603 v4	
Intel®	Xeon®	E5-2608L v4	
Intel®	Xeon®	E5-2609 v4	
Intel®	Xeon®	E5-2618L v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-2620 v4	
Intel®	Xeon®	E5-2623 v4	
Intel®	Xeon®	E5-2628L v4	
Intel®	Xeon®	E5-2630 v4	
Intel®	Xeon®	E5-2630L v4	
Intel®	Xeon®	E5-2637 v4	
Intel®	Xeon®	E5-2640 v4	
Intel®	Xeon®	E5-2643 v4	
Intel®	Xeon®	E5-2648L v4	
Intel®	Xeon®	E5-2650 v4	
Intel®	Xeon®	E5-2650L v4	
Intel®	Xeon®	E5-2658 v4	
Intel®	Xeon®	E5-2660 v4	
Intel®	Xeon®	E5-2667 v4	
Intel®	Xeon®	E5-2680 v4	
Intel®	Xeon®	E5-2683 v4	
Intel®	Xeon®	E5-2687W v4	
Intel®	Xeon®	E5-2690 v4	
Intel®	Xeon®	E5-2695 v4	
Intel®	Xeon®	E5-2697 v4	
Intel®	Xeon®	E5-2697A v4	
Intel®	Xeon®	E5-2698 v4	
Intel®	Xeon®	E5-2699 v4	
Intel®	Xeon®	E5-2699A v4	
Intel®	Xeon®	E5-2699R v4	
Intel®	Xeon®	E5-4610 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-4620 v4	
Intel®	Xeon®	E5-4627 v4	
Intel®	Xeon®	E5-4628L v4	
Intel®	Xeon®	E5-4640 v4	
Intel®	Xeon®	E5-4650 v4	
Intel®	Xeon®	E5-4655 v4	
Intel®	Xeon®	E5-4660 v4	
Intel®	Xeon®	E5-4667 v4	
Intel®	Xeon®	E5-4669 v4	
Intel®	Xeon®	E7-4809 v4	
Intel®	Xeon®	E7-4820 v4	
Intel®	Xeon®	E7-4830 v4	
Intel®	Xeon®	E7-4850 v4	
Intel®	Xeon®	E7-8860 v4	
Intel®	Xeon®	E7-8867 v4	
Intel®	Xeon®	E7-8870 v4	
Intel®	Xeon®	E7-8880 v4	
Intel®	Xeon®	E7-8890 v4	
Intel®	Xeon®	E7-8891 v4	
Intel®	Xeon®	E7-8893 v4	
Intel®	Xeon®	E7-8894 v4	
Intel®	Xeon®	Gold 5115	
Intel®	Xeon®	Gold 5118	
Intel®	Xeon®	Gold 5119T	
Intel®	Xeon®	Gold 5120	
Intel®	Xeon®	Gold 5120T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 5122	
Intel®	Xeon®	Gold 5215	
Intel®	Xeon®	Gold 5215L	
Intel®	Xeon®	Gold 5217	
Intel®	Xeon®	Gold 5218B	
Intel®	Xeon®	Gold 5218N	
Intel®	Xeon®	Gold 5218R	
Intel®	Xeon®	Gold 5218T	
Intel®	Xeon®	Gold 5220	
Intel®	Xeon®	Gold 5220R	
Intel®	Xeon®	Gold 5220S	
Intel®	Xeon®	Gold 5220T	
Intel®	Xeon®	Gold 5222	
Intel®	Xeon®	Gold 5315Y	
Intel®	Xeon®	Gold 5317	
Intel®	Xeon®	Gold 5318N	
Intel®	Xeon®	Gold 5318S	
Intel®	Xeon®	Gold 5320	
Intel®	Xeon®	Gold 5320T	
Intel®	Xeon®	Gold 6126	
Intel®	Xeon®	Gold 6126F	
Intel®	Xeon®	Gold 6126T	
Intel®	Xeon®	Gold 6128	
Intel®	Xeon®	Gold 6130	
Intel®	Xeon®	Gold 6130F	
Intel®	Xeon®	Gold 6130T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6132	
Intel®	Xeon®	Gold 6134	
Intel®	Xeon®	Gold 6136	
Intel®	Xeon®	Gold 6138	
Intel®	Xeon®	Gold 6138F	
Intel®	Xeon®	Gold 6138P	
Intel®	Xeon®	Gold 6138T	
Intel®	Xeon®	Gold 6140	
Intel®	Xeon®	Gold 6142	
Intel®	Xeon®	Gold 6142F	
Intel®	Xeon®	Gold 6144	
Intel®	Xeon®	Gold 6146	
Intel®	Xeon®	Gold 6148	
Intel®	Xeon®	Gold 6148F	
Intel®	Xeon®	Gold 6150	
Intel®	Xeon®	Gold 6152	
Intel®	Xeon®	Gold 6154	
Intel®	Xeon®	Gold 6208U	
Intel®	Xeon®	Gold 6209U	
Intel®	Xeon®	Gold 6210U	
Intel®	Xeon®	Gold 6212U	
Intel®	Xeon®	Gold 6222V	
Intel®	Xeon®	Gold 6226	
Intel®	Xeon®	Gold 6226R	
Intel®	Xeon®	Gold 6230	
Intel®	Xeon®	Gold 6230N	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6230R	
Intel®	Xeon®	Gold 6230T	
Intel®	Xeon®	Gold 6238	
Intel®	Xeon®	Gold 6238L	
Intel®	Xeon®	Gold 6238T	
Intel®	Xeon®	Gold 6240	
Intel®	Xeon®	Gold 6240L	
Intel®	Xeon®	Gold 6240R	
Intel®	Xeon®	Gold 6240Y	
Intel®	Xeon®	Gold 6242	
Intel®	Xeon®	Gold 6242R	
Intel®	Xeon®	Gold 6244	
Intel®	Xeon®	Gold 6246R	
Intel®	Xeon®	Gold 6248	
Intel®	Xeon®	Gold 6248R	
Intel®	Xeon®	Gold 6250	
Intel®	Xeon®	Gold 6250L	
Intel®	Xeon®	Gold 6252	
Intel®	Xeon®	Gold 6252N	
Intel®	Xeon®	Gold 6254	
Intel®	Xeon®	Gold 6256	
Intel®	Xeon®	Gold 6258R	
Intel®	Xeon®	Gold 6262V	
Intel®	Xeon®	Gold 6312U	
Intel®	Xeon®	Gold 6314U	
Intel®	Xeon®	Gold 6326	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6330	
Intel®	Xeon®	Gold 6330N	
Intel®	Xeon®	Gold 6334	
Intel®	Xeon®	Gold 6336Y	
Intel®	Xeon®	Gold 6338	
Intel®	Xeon®	Gold 6338N	
Intel®	Xeon®	Gold 6338T	
Intel®	Xeon®	Gold 6342	
Intel®	Xeon®	Gold 6346	
Intel®	Xeon®	Gold 6348	
Intel®	Xeon®	Gold 6354	
Intel®	Xeon®	Gold Gold 5218	
Intel®	Xeon®	Gold Gold 5318Y	
Intel®	Xeon®	Gold Gold 6238R	
Intel®	Xeon®	Gold6246	
Intel®	Xeon®	Goldv 6234	
Intel®	Xeon®	Platinum 8153	
Intel®	Xeon®	Platinum 8156	
Intel®	Xeon®	Platinum 8158	
Intel®	Xeon®	Platinum 8160	
Intel®	Xeon®	Platinum 8160F	
Intel®	Xeon®	Platinum 8160T	
Intel®	Xeon®	Platinum 8164	
Intel®	Xeon®	Platinum 8168	
Intel®	Xeon®	Platinum 8170	
Intel®	Xeon®	Platinum 8176	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Platinum 8176F	
Intel®	Xeon®	Platinum 8180	
Intel®	Xeon®	Platinum 8253	
Intel®	Xeon®	Platinum 8256	
Intel®	Xeon®	Platinum 8260	
Intel®	Xeon®	Platinum 8260L	
Intel®	Xeon®	Platinum 8260Y	
Intel®	Xeon®	Platinum 8268	
Intel®	Xeon®	Platinum 8270	
Intel®	Xeon®	Platinum 8276	
Intel®	Xeon®	Platinum 8276L	
Intel®	Xeon®	Platinum 8280	
Intel®	Xeon®	Platinum 8280L	
Intel®	Xeon®	Platinum 8351N	
Intel®	Xeon®	Platinum 8352S	
Intel®	Xeon®	Platinum 8352V	
Intel®	Xeon®	Platinum 8352Y	
Intel®	Xeon®	Platinum 8358	
Intel®	Xeon®	Platinum 8358P	
Intel®	Xeon®	Platinum 8360Y	
Intel®	Xeon®	Platinum 8368	
Intel®	Xeon®	Platinum 8368Q	
Intel®	Xeon®	Platinum 8380	
Intel®	Xeon®	Platinum 9221	
Intel®	Xeon®	Platinum 9222	
Intel®	Xeon®	Platinum 9242	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Platinum 9282	
Intel®	Xeon®	Silver 4108	
Intel®	Xeon®	Silver 4109T	
Intel®	Xeon®	Silver 4110	
Intel®	Xeon®	Silver 4112	
Intel®	Xeon®	Silver 4114	
Intel®	Xeon®	Silver 4114T	
Intel®	Xeon®	Silver 4116	
Intel®	Xeon®	Silver 4116T	
Intel®	Xeon®	Silver 4208	
Intel®	Xeon®	Silver 4209T	
Intel®	Xeon®	Silver 4210	
Intel®	Xeon®	Silver 4210R	
Intel®	Xeon®	Silver 4210T	
Intel®	Xeon®	Silver 4214	
Intel®	Xeon®	Silver 4214R	
Intel®	Xeon®	Silver 4214Y	
Intel®	Xeon®	Silver 4215	
Intel®	Xeon®	Silver 4215R	
Intel®	Xeon®	Silver 4216	
Intel®	Xeon®	Silver 4309Y	
Intel®	Xeon®	Silver 4310	
Intel®	Xeon®	Silver 4310T	
Intel®	Xeon®	Silver 4314	
Intel®	Xeon®	Silver 4316	
Intel®	Xeon®	W-10855M	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-10885M	
Intel®	Xeon®	W-1250	
Intel®	Xeon®	W-1250E	
Intel®	Xeon®	W-1250P	
Intel®	Xeon®	W-1250TE	
Intel®	Xeon®	W-1270	
Intel®	Xeon®	W-1270E	
Intel®	Xeon®	W-1270P	
Intel®	Xeon®	W-1270TE	
Intel®	Xeon®	W-1290	
Intel®	Xeon®	W-1290E	
Intel®	Xeon®	W-1290P	
Intel®	Xeon®	W-1290T	
Intel®	Xeon®	W-1290TE	
Intel®	Xeon®	W-2123	
Intel®	Xeon®	W-2125	
Intel®	Xeon®	W-2133	
Intel®	Xeon®	W-2135	
Intel®	Xeon®	W-2145	
Intel®	Xeon®	W-2155	
Intel®	Xeon®	W-2175	
Intel®	Xeon®	W-2195	
Intel®	Xeon®	W-2223	
Intel®	Xeon®	W-2225	
Intel®	Xeon®	W-2235	
Intel®	Xeon®	W-2245	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-2255	
Intel®	Xeon®	W-2265	
Intel®	Xeon®	W-2275	
Intel®	Xeon®	W-2295	
Intel®	Xeon®	W-3175X	
Intel®	Xeon®	W-3223	
Intel®	Xeon®	W-3225	
Intel®	Xeon®	W-3235	
Intel®	Xeon®	W-3245	
Intel®	Xeon®	W-3245M	
Intel®	Xeon®	W-3265	
Intel®	Xeon®	W-3265M	
Intel®	Xeon®	W-3275	
Intel®	Xeon®	W-3275M	

Windows 10 21H1 Supported Qualcomm Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	850
Qualcomm®	Snapdragon™	8cx

Windows 10 21H2 Supported AMD Processors

Article • 05/23/2023

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	AMD	3015e
AMD	AMD	3020e
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution

Manufacturer	Brand	Model
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G

Manufacturer	Brand	Model
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G
AMD	Athlon™	Gold 3150GE
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e

Manufacturer	Brand	Model
AMD	Athlon™	Silver 3050GE
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7313
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	7401
AMD	EPYC™	7402
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453

Manufacturer	Brand	Model
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7551
AMD	EPYC™	7552
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7713
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7232P
AMD	EPYC™	72F3
AMD	EPYC™	7302P
AMD	EPYC™	7313P
AMD	EPYC™	7351P
AMD	EPYC™	73F3
AMD	EPYC™	7401P
AMD	EPYC™	7402P
AMD	EPYC™	7443P

Manufacturer	Brand	Model
AMD	EPYC™	74F3
AMD	EPYC™	7502P
AMD	EPYC™	7543P
AMD	EPYC™	7551P
AMD	EPYC™	75F3
AMD	EPYC™	7702P
AMD	EPYC™	7713P
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100

Manufacturer	Brand	Model
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590

Manufacturer	Brand	Model
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	3100
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics

Manufacturer	Brand	Model
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G
AMD	Ryzen™ 3	4300GE
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3	5300U
AMD	Ryzen™ 3	5400U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U

Manufacturer	Brand	Model
AMD	Ryzen™ 3 PRO	5350G
AMD	Ryzen™ 3 PRO	5350GE
AMD	Ryzen™ 3 PRO	5450U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600X

Manufacturer	Brand	Model
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G
AMD	Ryzen™ 5	4600GE
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5	5300G
AMD	Ryzen™ 5	5300GE
AMD	Ryzen™ 5	5500U
AMD	Ryzen™ 5	5600G
AMD	Ryzen™ 5	5600GE
AMD	Ryzen™ 5	5600H
AMD	Ryzen™ 5	5600HS
AMD	Ryzen™ 5	5600U
AMD	Ryzen™ 5	5600X
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	4650G

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 5 PRO	5650G
AMD	Ryzen™ 5 PRO	5650GE
AMD	Ryzen™ 5 PRO	5650U
AMD	Ryzen™ 5 PRO	5750G
AMD	Ryzen™ 5 PRO	5750GE
AMD	Ryzen™ 7	1700
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G
AMD	Ryzen™ 7	4700GE
AMD	Ryzen™ 7	4700U

Manufacturer	Brand	Model
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7	5700G
AMD	Ryzen™ 7	5700GE
AMD	Ryzen™ 7	5700U
AMD	Ryzen™ 7	5800H
AMD	Ryzen™ 7	5800HS
AMD	Ryzen™ 7	5800U
AMD	Ryzen™ 7	5800X
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	1700X
AMD	Ryzen™ 7 PRO	2700U
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 7 PRO	5850U
AMD	Ryzen™ 9	5900
AMD	Ryzen™ 9	3900 Processor
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H

Manufacturer	Brand	Model
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9	5900HS
AMD	Ryzen™ 9	5900HX
AMD	Ryzen™ 9	5900X
AMD	Ryzen™ 9	5950X
AMD	Ryzen™ 9	5980HS
AMD	Ryzen™ 9	5980HX
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 21H2 Supported Intel Processors

Article • 02/26/2024

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Intel®	Atom®	E3805	IoT Enterprise Only
Intel®	Atom®	E3815	IoT Enterprise Only
Intel®	Atom®	E3825	IoT Enterprise Only
Intel®	Atom®	E3826	IoT Enterprise Only
Intel®	Atom®	E3827	IoT Enterprise Only
Intel®	Atom®	E3845	IoT Enterprise Only
Intel®	Atom®	x5-E3930	IoT Enterprise Only
Intel®	Atom®	x5-E3940	IoT Enterprise Only
Intel®	Atom®	x5-E3950	IoT Enterprise Only
Intel®	Atom®	x5-E8000	IoT Enterprise Only
Intel®	Atom®	x6200FE	
Intel®	Atom®	x6211E	
Intel®	Atom®	x6212RE	
Intel®	Atom®	x6413E	
Intel®	Atom®	x6414RE	
Intel®	Atom®	x6425E	
Intel®	Atom®	x6425RE	
Intel®	Atom®	x6427FE	
Intel®	Celeron®	6305	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	3205U	
Intel®	Celeron®	3215U	
Intel®	Celeron®	3755U	
Intel®	Celeron®	3765U	
Intel®	Celeron®	3855U	
Intel®	Celeron®	3865U	
Intel®	Celeron®	3867U	
Intel®	Celeron®	3955U	
Intel®	Celeron®	3965U	
Intel®	Celeron®	3965Y	
Intel®	Celeron®	4205U	
Intel®	Celeron®	4305U	
Intel®	Celeron®	4305UE	
Intel®	Celeron®	5205U	
Intel®	Celeron®	5305U	
Intel®	Celeron®	6305E	
Intel®	Celeron®	7305E	IoT Enterprise Only
Intel®	Celeron®	G3900	
Intel®	Celeron®	G3900E	
Intel®	Celeron®	G3900T	
Intel®	Celeron®	G3900TE	
Intel®	Celeron®	G3902E	
Intel®	Celeron®	G3920	
Intel®	Celeron®	G3930	
Intel®	Celeron®	G3930E	
Intel®	Celeron®	G3930T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	G3930TE	
Intel®	Celeron®	G3950	
Intel®	Celeron®	G4900	
Intel®	Celeron®	G4900T	
Intel®	Celeron®	G4920	
Intel®	Celeron®	G4930	
Intel®	Celeron®	G4930E	
Intel®	Celeron®	G4930T	
Intel®	Celeron®	G4932E	
Intel®	Celeron®	G4950	
Intel®	Celeron®	G5900	
Intel®	Celeron®	G5900E	
Intel®	Celeron®	G5900T	
Intel®	Celeron®	G5900TE	
Intel®	Celeron®	G5905	
Intel®	Celeron®	G5905T	
Intel®	Celeron®	G5920	
Intel®	Celeron®	G5925	
Intel®	Celeron®	G6900E	IoT Enterprise Only
Intel®	Celeron®	G6900TE	IoT Enterprise Only
Intel®	Celeron®	J1900	IoT Enterprise Only
Intel®	Celeron®	J3355	IoT Enterprise Only
Intel®	Celeron®	J3355E	IoT Enterprise Only
Intel®	Celeron®	J3455	IoT Enterprise Only
Intel®	Celeron®	J3455E	IoT Enterprise Only
Intel®	Celeron®	J4005	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	J4025	
Intel®	Celeron®	J4105	
Intel®	Celeron®	J4115	
Intel®	Celeron®	J4125	
Intel®	Celeron®	J6412	
Intel®	Celeron®	J6413	
Intel®	Celeron®	N4000	
Intel®	Celeron®	N4020	
Intel®	Celeron®	N4100	
Intel®	Celeron®	N4120	
Intel®	Celeron®	N4500	
Intel®	Celeron®	N4505	
Intel®	Celeron®	N5100	
Intel®	Celeron®	N5105	
Intel®	Celeron®	N6210	
Intel®	Celeron®	N6211	
Intel®	Core™	3-100U	new
Intel®	Core™	3-100U with IPU	new
Intel®	Core™	5-120U	new
Intel®	Core™	7-150U	new
Intel®	Core™	i3-1000G1	
Intel®	Core™	i3-1000G4	
Intel®	Core™	i3-1005G1	
Intel®	Core™	i3-10100	
Intel®	Core™	i3-10100E	
Intel®	Core™	i3-10100F	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-10100T	
Intel®	Core™	i3-10100TE	
Intel®	Core™	i3-10100Y	
Intel®	Core™	i3-10105	
Intel®	Core™	i3-10105F	
Intel®	Core™	i3-10105T	
Intel®	Core™	i3-10110U	
Intel®	Core™	i3-10110Y	
Intel®	Core™	i3-10300	
Intel®	Core™	i3-10300T	
Intel®	Core™	i3-10305	
Intel®	Core™	i3-10305T	
Intel®	Core™	i3-10320	
Intel®	Core™	i3-10325	
Intel®	Core™	i3-1110G4	
Intel®	Core™	i3-1115G4	
Intel®	Core™	i3-1115G4E	
Intel®	Core™	i3-1115GRE	
Intel®	Core™	i3-1120G4	
Intel®	Core™	i3-1125G4	
Intel®	Core™	i3-12100	
Intel®	Core™	i3-12100E	IoT Enterprise Only
Intel®	Core™	i3-12100TE	IoT Enterprise Only
Intel®	Core™	i3-1215UE	IoT Enterprise Only
Intel®	Core™	i3-1220PE	IoT Enterprise Only
Intel®	Core™	i3-12300HE	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-14100	new
Intel®	Core™	i3-14100F	new
Intel®	Core™	i3-14100T	new
Intel®	Core™	i3-5005U	
Intel®	Core™	i3-5010U	
Intel®	Core™	i3-5015U	
Intel®	Core™	i3-5020U	
Intel®	Core™	i3-5157U	
Intel®	Core™	i3-6006U	
Intel®	Core™	i3-6098P	
Intel®	Core™	i3-6100	
Intel®	Core™	i3-6100E	
Intel®	Core™	i3-6100H	
Intel®	Core™	i3-6100T	
Intel®	Core™	i3-6100TE	
Intel®	Core™	i3-6100U	
Intel®	Core™	i3-6102E	
Intel®	Core™	i3-6157U	
Intel®	Core™	i3-6167U	
Intel®	Core™	i3-6300	
Intel®	Core™	i3-6300T	
Intel®	Core™	i3-6320	
Intel®	Core™	i3-7020U	
Intel®	Core™	i3-7100	
Intel®	Core™	i3-7100E	
Intel®	Core™	i3-7100H	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-7100T	
Intel®	Core™	i3-7100U	
Intel®	Core™	i3-7101E	
Intel®	Core™	i3-7101TE	
Intel®	Core™	i3-7102E	
Intel®	Core™	i3-7130U	
Intel®	Core™	i3-7167U	
Intel®	Core™	i3-7300	
Intel®	Core™	i3-7300T	
Intel®	Core™	i3-7320	
Intel®	Core™	i3-7350K	
Intel®	Core™	i3-8100	
Intel®	Core™	i3-8100B	
Intel®	Core™	i3-8100H	
Intel®	Core™	i3-8100T	
Intel®	Core™	i3-8109U	
Intel®	Core™	i3-8130U	
Intel®	Core™	i3-8140U	
Intel®	Core™	i3-8145U	
Intel®	Core™	i3-8145UE	
Intel®	Core™	i3-8300	
Intel®	Core™	i3-8300T	
Intel®	Core™	i3-8350K	
Intel®	Core™	i3-9100	
Intel®	Core™	i3-9100E	
Intel®	Core™	i3-9100F	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-9100HL	
Intel®	Core™	i3-9100T	
Intel®	Core™	i3-9100TE	
Intel®	Core™	i3-9300	
Intel®	Core™	i3-9300T	
Intel®	Core™	i3-9320	
Intel®	Core™	i3-9350K	
Intel®	Core™	i3-9350KF	
Intel®	Core™	i3-L13G4	
Intel®	Core™	i5-10200H	
Intel®	Core™	i5-10210U	
Intel®	Core™	i5-10210Y	
Intel®	Core™	i5-10300H	
Intel®	Core™	i5-1030G4	
Intel®	Core™	i5-1030G7	
Intel®	Core™	i5-10310U	
Intel®	Core™	i5-10310Y	
Intel®	Core™	i5-1035G1	
Intel®	Core™	i5-1035G4	
Intel®	Core™	i5-1035G7	
Intel®	Core™	i5-1038NG7	
Intel®	Core™	i5-10400	
Intel®	Core™	i5-10400F	
Intel®	Core™	i5-10400H	
Intel®	Core™	i5-10400T	
Intel®	Core™	i5-10500	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-10500E	
Intel®	Core™	i5-10500H	
Intel®	Core™	i5-10500T	
Intel®	Core™	i5-10500TE	
Intel®	Core™	i5-10505	
Intel®	Core™	i5-10600	
Intel®	Core™	i5-10600K	
Intel®	Core™	i5-10600KF	
Intel®	Core™	i5-10600T	
Intel®	Core™	i5-11260H	
Intel®	Core™	i5-11300H	
Intel®	Core™	i5-1130G7	
Intel®	Core™	i5-11320H	
Intel®	Core™	i5-1135G7	
Intel®	Core™	i5-11400	
Intel®	Core™	i5-11400F	
Intel®	Core™	i5-11400H	
Intel®	Core™	i5-11400T	
Intel®	Core™	i5-1140G7	
Intel®	Core™	i5-1145G7	
Intel®	Core™	i5-1145G7E	
Intel®	Core™	i5-1145GRE	
Intel®	Core™	i5-11500	
Intel®	Core™	i5-11500H	
Intel®	Core™	i5-11500T	
Intel®	Core™	i5-1155G7	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-11600	
Intel®	Core™	i5-11600K	
Intel®	Core™	i5-11600KF	
Intel®	Core™	i5-11600T	
Intel®	Core™	i5-1235U	
Intel®	Core™	i5-12400	
Intel®	Core™	i5-1245UE	IoT Enterprise Only
Intel®	Core™	i5-12500	
Intel®	Core™	i5-12500E	IoT Enterprise Only
Intel®	Core™	i5-12500TE	IoT Enterprise Only
Intel®	Core™	i5-1250PE	IoT Enterprise Only
Intel®	Core™	i5-12600HE	IoT Enterprise Only
Intel®	Core™	i5-12600K	
Intel®	Core™	i5-12600KF	
Intel®	Core™	i5-14400	new
Intel®	Core™	i5-14400F	new
Intel®	Core™	i5-14400T	new
Intel®	Core™	i5-14500	new
Intel®	Core™	i5-14500T	new
Intel®	Core™	i5-14600	new
Intel®	Core™	i5-14600K	new
Intel®	Core™	i5-14600KF	new
Intel®	Core™	i5-14600T	new
Intel®	Core™	i5-5200U	
Intel®	Core™	i5-5250U	
Intel®	Core™	i5-5257U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-5287U	
Intel®	Core™	i5-5300U	
Intel®	Core™	i5-5350H	
Intel®	Core™	i5-5350U	
Intel®	Core™	i5-5575R	
Intel®	Core™	i5-5675C	
Intel®	Core™	i5-5675R	
Intel®	Core™	i5-6200U	
Intel®	Core™	i5-6260U	
Intel®	Core™	i5-6267U	
Intel®	Core™	i5-6287U	
Intel®	Core™	i5-6300HQ	
Intel®	Core™	i5-6300U	
Intel®	Core™	i5-6350HQ	
Intel®	Core™	i5-6360U	
Intel®	Core™	i5-6400	
Intel®	Core™	i5-6400T	
Intel®	Core™	i5-6402P	
Intel®	Core™	i5-6440EQ	
Intel®	Core™	i5-6440HQ	
Intel®	Core™	i5-6442EQ	
Intel®	Core™	i5-6500	
Intel®	Core™	i5-6500T	
Intel®	Core™	i5-6500TE	
Intel®	Core™	i5-6585R	
Intel®	Core™	i5-6600	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-6600K	
Intel®	Core™	i5-6600T	
Intel®	Core™	i5-6685R	
Intel®	Core™	i5-7200U	
Intel®	Core™	i5-7260U	
Intel®	Core™	i5-7267U	
Intel®	Core™	i5-7287U	
Intel®	Core™	i5-7300HQ	
Intel®	Core™	i5-7300U	
Intel®	Core™	i5-7360U	
Intel®	Core™	i5-7400	
Intel®	Core™	i5-7400T	
Intel®	Core™	i5-7440EQ	
Intel®	Core™	i5-7440HQ	
Intel®	Core™	i5-7442EQ	
Intel®	Core™	i5-7500	
Intel®	Core™	i5-7500T	
Intel®	Core™	i5-7600	
Intel®	Core™	i5-7600K	
Intel®	Core™	i5-7600T	
Intel®	Core™	i5-7640X	
Intel®	Core™	i5-7Y54	
Intel®	Core™	i5-7Y57	
Intel®	Core™	i5-8200Y	
Intel®	Core™	i5-8210Y	
Intel®	Core™	i5-8250U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-8257U	
Intel®	Core™	i5-8259U	
Intel®	Core™	i5-8260U	
Intel®	Core™	i5-8265U	
Intel®	Core™	i5-8269U	
Intel®	Core™	i5-8279U	
Intel®	Core™	i5-8300H	
Intel®	Core™	i5-8305G	
Intel®	Core™	i5-8310Y	
Intel®	Core™	i5-8350U	
Intel®	Core™	i5-8365U	
Intel®	Core™	i5-8365UE	
Intel®	Core™	i5-8400	
Intel®	Core™	i5-8400B	
Intel®	Core™	i5-8400H	
Intel®	Core™	i5-8400T	
Intel®	Core™	i5-8500	
Intel®	Core™	i5-8500B	
Intel®	Core™	i5-8500T	
Intel®	Core™	i5-8600	
Intel®	Core™	i5-8600K	
Intel®	Core™	i5-8600T	
Intel®	Core™	i5-9300H	
Intel®	Core™	i5-9300HF	
Intel®	Core™	i5-9400	
Intel®	Core™	i5-9400F	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-9400H	
Intel®	Core™	i5-9400T	
Intel®	Core™	i5-9500	
Intel®	Core™	i5-9500E	
Intel®	Core™	i5-9500F	
Intel®	Core™	i5-9500T	
Intel®	Core™	i5-9500TE	
Intel®	Core™	i5-9600	
Intel®	Core™	i5-9600K	
Intel®	Core™	i5-9600KF	
Intel®	Core™	i5-9600T	
Intel®	Core™	i5-L16G7	
Intel®	Core™	i7-10510U	
Intel®	Core™	i7-10510Y	
Intel®	Core™	i7-1060G7	
Intel®	Core™	i7-10610U	
Intel®	Core™	i7-1065G7	
Intel®	Core™	i7-1068NG7	
Intel®	Core™	i7-10700	
Intel®	Core™	i7-10700E	
Intel®	Core™	i7-10700F	
Intel®	Core™	i7-10700K	
Intel®	Core™	i7-10700KF	
Intel®	Core™	i7-10700T	
Intel®	Core™	i7-10700TE	
Intel®	Core™	i7-10710U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-10750H	
Intel®	Core™	i7-10810U	
Intel®	Core™	i7-10850H	
Intel®	Core™	i7-10870H	
Intel®	Core™	i7-10875H	
Intel®	Core™	i7-11370H	
Intel®	Core™	i7-11375H	
Intel®	Core™	i7-11390H	
Intel®	Core™	i7-11600H	
Intel®	Core™	i7-1160G7	
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-11700	
Intel®	Core™	i7-11700F	
Intel®	Core™	i7-11700K	
Intel®	Core™	i7-11700KF	
Intel®	Core™	i7-11700T	
Intel®	Core™	i7-11800H	
Intel®	Core™	i7-1180G7	
Intel®	Core™	i7-11850H	
Intel®	Core™	i7-1185G7	
Intel®	Core™	i7-1185G7E	
Intel®	Core™	i7-1185GRE	
Intel®	Core™	i7-1195G7	
Intel®	Core™	i7-1265UE	IoT Enterprise Only
Intel®	Core™	i7-12700	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-12700E	IoT Enterprise Only
Intel®	Core™	i7-12700K	
Intel®	Core™	i7-12700KF	
Intel®	Core™	i7-12700TE	IoT Enterprise Only
Intel®	Core™	i7-1270PE	IoT Enterprise Only
Intel®	Core™	i7-12800HE	IoT Enterprise Only
Intel®	Core™	i7-14700	new
Intel®	Core™	i7-14700F	new
Intel®	Core™	i7-14700K	new
Intel®	Core™	i7-14700KF	new
Intel®	Core™	i7-14700T	new
Intel®	Core™	i7-5500U	
Intel®	Core™	i7-5550U	
Intel®	Core™	i7-5557U	
Intel®	Core™	i7-5600U	
Intel®	Core™	i7-5650U	
Intel®	Core™	i7-5700EQ	
Intel®	Core™	i7-5700HQ	
Intel®	Core™	i7-5750HQ	
Intel®	Core™	i7-5775C	
Intel®	Core™	i7-5775R	
Intel®	Core™	i7-5850EQ	
Intel®	Core™	i7-5850HQ	
Intel®	Core™	i7-5950HQ	
Intel®	Core™	i7-6500U	
Intel®	Core™	i7-6560U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-6567U	
Intel®	Core™	i7-6600U	
Intel®	Core™	i7-6650U	
Intel®	Core™	i7-6660U	
Intel®	Core™	i7-6700	
Intel®	Core™	i7-6700HQ	
Intel®	Core™	i7-6700K	
Intel®	Core™	i7-6700T	
Intel®	Core™	i7-6700TE	
Intel®	Core™	i7-6770HQ	
Intel®	Core™	i7-6785R	
Intel®	Core™	i7-6820EQ	
Intel®	Core™	i7-6820HK	
Intel®	Core™	i7-6820HQ	
Intel®	Core™	i7-6822EQ	
Intel®	Core™	i7-6870HQ	
Intel®	Core™	i7-6920HQ	
Intel®	Core™	i7-6970HQ	
Intel®	Core™	i7-7500U	
Intel®	Core™	i7-7560U	
Intel®	Core™	i7-7567U	
Intel®	Core™	i7-7600U	
Intel®	Core™	i7-7660U	
Intel®	Core™	i7-7700	
Intel®	Core™	i7-7700HQ	
Intel®	Core™	i7-7700K	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-7700T	
Intel®	Core™	i7-7740X	
Intel®	Core™	i7-7800X	
Intel®	Core™	i7-7820EQ	
Intel®	Core™	i7-7820HK	
Intel®	Core™	i7-7820HQ	
Intel®	Core™	i7-7820X	
Intel®	Core™	i7-7920HQ	
Intel®	Core™	i7-7Y75	
Intel®	Core™	i7-8086K	
Intel®	Core™	i7-8500Y	
Intel®	Core™	i7-8550U	
Intel®	Core™	i7-8557U	
Intel®	Core™	i7-8559U	
Intel®	Core™	i7-8565U	
Intel®	Core™	i7-8569U	
Intel®	Core™	i7-8650U	
Intel®	Core™	i7-8665U	
Intel®	Core™	i7-8665UE	
Intel®	Core™	i7-8700	
Intel®	Core™	i7-8700B	
Intel®	Core™	i7-8700K	
Intel®	Core™	i7-8700T	
Intel®	Core™	i7-8705G	
Intel®	Core™	i7-8706G	
Intel®	Core™	i7-8709G	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-8750H	
Intel®	Core™	i7-8809G	
Intel®	Core™	i7-8850H	
Intel®	Core™	i7-9700	
Intel®	Core™	i7-9700E	
Intel®	Core™	i7-9700F	
Intel®	Core™	i7-9700K	
Intel®	Core™	i7-9700KF	
Intel®	Core™	i7-9700T	
Intel®	Core™	i7-9700TE	
Intel®	Core™	i7-9750H	
Intel®	Core™	i7-9750HF	
Intel®	Core™	i7-9800X	
Intel®	Core™	i7-9850H	
Intel®	Core™	i7-9850HE	
Intel®	Core™	i7-9850HL	
Intel®	Core™	i9-10850K	
Intel®	Core™	i9-10885H	
Intel®	Core™	i9-10900	
Intel®	Core™	i9-10900E	
Intel®	Core™	i9-10900F	
Intel®	Core™	i9-10900K	
Intel®	Core™	i9-10900KF	
Intel®	Core™	i9-10900T	
Intel®	Core™	i9-10900TE	
Intel®	Core™	i9-10900X	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-10920X	
Intel®	Core™	i9-10940X	
Intel®	Core™	i9-10980HK	
Intel®	Core™	i9-10980XE	
Intel®	Core™	i9-11900	
Intel®	Core™	i9-11900F	
Intel®	Core™	i9-11900H	
Intel®	Core™	i9-11900K	
Intel®	Core™	i9-11900KF	
Intel®	Core™	i9-11900T	
Intel®	Core™	i9-11950H	
Intel®	Core™	i9-11980HK	
Intel®	Core™	i9-12900	
Intel®	Core™	i9-12900E	IoT Enterprise Only
Intel®	Core™	i9-12900K	
Intel®	Core™	i9-12900KF	
Intel®	Core™	i9-12900TE	IoT Enterprise Only
Intel®	Core™	i9-14900	new
Intel®	Core™	i9-14900F	new
Intel®	Core™	i9-14900K	new
Intel®	Core™	i9-14900KF	new
Intel®	Core™	i9-14900T	new
Intel®	Core™	i9-7900X	
Intel®	Core™	i9-7920X	
Intel®	Core™	i9-7940X	
Intel®	Core™	i9-7960X	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-7980XE	
Intel®	Core™	i9-8950HK	
Intel®	Core™	i9-9820X	
Intel®	Core™	i9-9880H	
Intel®	Core™	i9-9900	
Intel®	Core™	i9-9900K	
Intel®	Core™	i9-9900KF	
Intel®	Core™	i9-9900KS	
Intel®	Core™	i9-9900T	
Intel®	Core™	i9-9900X	
Intel®	Core™	i9-9920X	
Intel®	Core™	i9-9940X	
Intel®	Core™	i9-9960X	
Intel®	Core™	i9-9980HK	
Intel®	Core™	i9-9980XE	
Intel®	Core™	m3-6Y30	
Intel®	Core™	m3-7Y30	
Intel®	Core™	m3-7Y32	
Intel®	Core™	m3-8100Y	
Intel®	Core™	m5-6Y54	
Intel®	Core™	m5-6Y57	
Intel®	Core™	M-5Y10	
Intel®	Core™	M-5Y10a	
Intel®	Core™	M-5Y10c	
Intel®	Core™	M-5Y31	
Intel®	Core™	M-5Y51	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	M-5Y70	
Intel®	Core™	M-5Y71	
Intel®	Core™	m7-6Y75	
Intel®	Core™	Ultra 5-125H	new
Intel®	Core™	Ultra 5-125U	new
Intel®	Core™	Ultra 5-134U	new
Intel®	Core™	Ultra 5-135H	new
Intel®	Core™	Ultra 5-135U	new
Intel®	Core™	Ultra 7-155H	new
Intel®	Core™	Ultra 7-155U	new
Intel®	Core™	Ultra 7-164U	new
Intel®	Core™	Ultra 7-165H	new
Intel®	Core™	Ultra 7-165U	new
Intel®	Core™	Ultra 9-185H	new
Intel®	Pentium®	6805	
Intel®	Pentium®	3805U	
Intel®	Pentium®	3825U	
Intel®	Pentium®	4405U	
Intel®	Pentium®	4405Y	
Intel®	Pentium®	D1507	
Intel®	Pentium®	D1508	
Intel®	Pentium®	D1509	
Intel®	Pentium®	D1517	
Intel®	Pentium®	D1519	
Intel®	Pentium®	G4400	
Intel®	Pentium®	G4400T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	G4400TE	
Intel®	Pentium®	G4500	
Intel®	Pentium®	G4500T	
Intel®	Pentium®	G4520	
Intel®	Pentium®	G4560	
Intel®	Pentium®	G4560T	
Intel®	Pentium®	G4600	
Intel®	Pentium®	G4600T	
Intel®	Pentium®	G4620	
Intel®	Pentium®	Gold 4410Y	
Intel®	Pentium®	Gold 4415U	
Intel®	Pentium®	Gold 4415Y	
Intel®	Pentium®	Gold 4417U	
Intel®	Pentium®	Gold 4425Y	
Intel®	Pentium®	Gold 5405U	
Intel®	Pentium®	Gold 6405U	
Intel®	Pentium®	Gold 6500Y	
Intel®	Pentium®	Gold 7505	
Intel®	Pentium®	Gold G5400	
Intel®	Pentium®	Gold G5400T	
Intel®	Pentium®	Gold G5420	
Intel®	Pentium®	Gold G5420T	
Intel®	Pentium®	Gold G5500	
Intel®	Pentium®	Gold G5500T	
Intel®	Pentium®	Gold G5600	
Intel®	Pentium®	Gold G5600T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	Gold G5620	
Intel®	Pentium®	Gold G6400	
Intel®	Pentium®	Gold G6400E	
Intel®	Pentium®	Gold G6400T	
Intel®	Pentium®	Gold G6400TE	
Intel®	Pentium®	Gold G6405	
Intel®	Pentium®	Gold G6405T	
Intel®	Pentium®	Gold G6500	
Intel®	Pentium®	Gold G6500T	
Intel®	Pentium®	Gold G6505	
Intel®	Pentium®	Gold G6505T	
Intel®	Pentium®	Gold G6600	
Intel®	Pentium®	Gold G6605	
Intel®	Pentium®	Gold G7400E	IoT Enterprise Only
Intel®	Pentium®	Gold G7400TE	IoT Enterprise Only
Intel®	Pentium®	J6426	
Intel®	Pentium®	N3710	IoT Enterprise Only
Intel®	Pentium®	N4200	IoT Enterprise Only
Intel®	Pentium®	N6415	
Intel®	Pentium®	Silver J5005	
Intel®	Pentium®	Silver J5040	
Intel®	Pentium®	Silver N5000	
Intel®	Pentium®	Silver N5030	
Intel®	Pentium®	Silver N6000	
Intel®	Pentium®	Silver N6005	
Intel®	Xeon®	Bronze 3104	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Bronze 3106	
Intel®	Xeon®	Bronze 3204	
Intel®	Xeon®	Bronze 3206R	
Intel®	Xeon®	D-1513N	
Intel®	Xeon®	D-1518	
Intel®	Xeon®	D-1520	
Intel®	Xeon®	D-1521	
Intel®	Xeon®	D-1523N	
Intel®	Xeon®	D-1527	
Intel®	Xeon®	D-1528	
Intel®	Xeon®	D-1529	
Intel®	Xeon®	D-1531	
Intel®	Xeon®	D-1533N	
Intel®	Xeon®	D-1537	
Intel®	Xeon®	D-1539	
Intel®	Xeon®	D-1540	
Intel®	Xeon®	D-1541	
Intel®	Xeon®	D-1543N	
Intel®	Xeon®	D-1548	
Intel®	Xeon®	D-1553N	
Intel®	Xeon®	D-1557	
Intel®	Xeon®	D-1559	
Intel®	Xeon®	D-1567	
Intel®	Xeon®	D-1571	
Intel®	Xeon®	D-1577	
Intel®	Xeon®	D-2123IT	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	D-2141I	
Intel®	Xeon®	D-2142IT	
Intel®	Xeon®	D-2143IT	
Intel®	Xeon®	D-2145NT	
Intel®	Xeon®	D-2146NT	
Intel®	Xeon®	D-2161I	
Intel®	Xeon®	D-2163IT	
Intel®	Xeon®	D-2166NT	
Intel®	Xeon®	D-2173IT	
Intel®	Xeon®	D-2177NT	
Intel®	Xeon®	D-2183IT	
Intel®	Xeon®	D-2187NT	
Intel®	Xeon®	E-2124	
Intel®	Xeon®	E-2124G	
Intel®	Xeon®	E-2126G	
Intel®	Xeon®	E-2134	
Intel®	Xeon®	E-2136	
Intel®	Xeon®	E-2144G	
Intel®	Xeon®	E-2146G	
Intel®	Xeon®	E-2174G	
Intel®	Xeon®	E-2176G	
Intel®	Xeon®	E-2176M	
Intel®	Xeon®	E-2186G	
Intel®	Xeon®	E-2186M	
Intel®	Xeon®	E-2224	
Intel®	Xeon®	E-2224G	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2226G	
Intel®	Xeon®	E-2226GE	
Intel®	Xeon®	E-2234	
Intel®	Xeon®	E-2236	
Intel®	Xeon®	E-2244G	
Intel®	Xeon®	E-2246G	
Intel®	Xeon®	E-2254ME	
Intel®	Xeon®	E-2254ML	
Intel®	Xeon®	E-2274G	
Intel®	Xeon®	E-2276G	
Intel®	Xeon®	E-2276M	
Intel®	Xeon®	E-2276ME	
Intel®	Xeon®	E-2276ML	
Intel®	Xeon®	E-2278G	
Intel®	Xeon®	E-2278GE	
Intel®	Xeon®	E-2278GEL	
Intel®	Xeon®	E-2286G	
Intel®	Xeon®	E-2286M	
Intel®	Xeon®	E-2288G	
Intel®	Xeon®	E3-1220 v5	
Intel®	Xeon®	E3-1220 v6	
Intel®	Xeon®	E3-1225 v5	
Intel®	Xeon®	E3-1225 v6	
Intel®	Xeon®	E3-1230 v5	
Intel®	Xeon®	E3-1230 v6	
Intel®	Xeon®	E3-1235L v5	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1240 v5	
Intel®	Xeon®	E3-1240 v6	
Intel®	Xeon®	E3-1240L v5	
Intel®	Xeon®	E3-1245 v5	
Intel®	Xeon®	E3-1245 v6	
Intel®	Xeon®	E3-1258L v4	
Intel®	Xeon®	E3-1260L v5	
Intel®	Xeon®	E3-1265L v4	
Intel®	Xeon®	E3-1268L v5	
Intel®	Xeon®	E3-1270 v5	
Intel®	Xeon®	E3-1270 v6	
Intel®	Xeon®	E3-1275 v5	
Intel®	Xeon®	E3-1275 v6	
Intel®	Xeon®	E3-1278L v4	
Intel®	Xeon®	E3-1280 v5	
Intel®	Xeon®	E3-1280 v6	
Intel®	Xeon®	E3-1285 v4	
Intel®	Xeon®	E3-1285 v6	
Intel®	Xeon®	E3-1285L v4	
Intel®	Xeon®	E3-1501L v6	
Intel®	Xeon®	E3-1501M v6	
Intel®	Xeon®	E3-1505L v5	
Intel®	Xeon®	E3-1505L v6	
Intel®	Xeon®	E3-1505M v5	
Intel®	Xeon®	E3-1505M v6	
Intel®	Xeon®	E3-1515M v5	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1535M v5	
Intel®	Xeon®	E3-1535M v6	
Intel®	Xeon®	E3-1545M v5	
Intel®	Xeon®	E3-1558L v5	
Intel®	Xeon®	E3-1565L v5	
Intel®	Xeon®	E3-1575M v5	
Intel®	Xeon®	E3-1578L v5	
Intel®	Xeon®	E3-1585 v5	
Intel®	Xeon®	E3-1585L v5	
Intel®	Xeon®	E5-1620 v4	
Intel®	Xeon®	E5-1630 v4	
Intel®	Xeon®	E5-1650 v4	
Intel®	Xeon®	E5-1660 v4	
Intel®	Xeon®	E5-1680 v4	
Intel®	Xeon®	E5-2603 v4	
Intel®	Xeon®	E5-2608L v4	
Intel®	Xeon®	E5-2609 v4	
Intel®	Xeon®	E5-2618L v4	
Intel®	Xeon®	E5-2620 v4	
Intel®	Xeon®	E5-2623 v4	
Intel®	Xeon®	E5-2628L v4	
Intel®	Xeon®	E5-2630 v4	
Intel®	Xeon®	E5-2630L v4	
Intel®	Xeon®	E5-2637 v4	
Intel®	Xeon®	E5-2640 v4	
Intel®	Xeon®	E5-2643 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-2648L v4	
Intel®	Xeon®	E5-2650 v4	
Intel®	Xeon®	E5-2650L v4	
Intel®	Xeon®	E5-2658 v4	
Intel®	Xeon®	E5-2660 v4	
Intel®	Xeon®	E5-2667 v4	
Intel®	Xeon®	E5-2680 v4	
Intel®	Xeon®	E5-2683 v4	
Intel®	Xeon®	E5-2687W v4	
Intel®	Xeon®	E5-2690 v4	
Intel®	Xeon®	E5-2695 v4	
Intel®	Xeon®	E5-2697 v4	
Intel®	Xeon®	E5-2697A v4	
Intel®	Xeon®	E5-2698 v4	
Intel®	Xeon®	E5-2699 v4	
Intel®	Xeon®	E5-2699A v4	
Intel®	Xeon®	E5-2699R v4	
Intel®	Xeon®	E5-4610 v4	
Intel®	Xeon®	E5-4620 v4	
Intel®	Xeon®	E5-4627 v4	
Intel®	Xeon®	E5-4628L v4	
Intel®	Xeon®	E5-4640 v4	
Intel®	Xeon®	E5-4650 v4	
Intel®	Xeon®	E5-4655 v4	
Intel®	Xeon®	E5-4660 v4	
Intel®	Xeon®	E5-4667 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-4669 v4	
Intel®	Xeon®	E7-4809 v4	
Intel®	Xeon®	E7-4820 v4	
Intel®	Xeon®	E7-4830 v4	
Intel®	Xeon®	E7-4850 v4	
Intel®	Xeon®	E7-8860 v4	
Intel®	Xeon®	E7-8867 v4	
Intel®	Xeon®	E7-8870 v4	
Intel®	Xeon®	E7-8880 v4	
Intel®	Xeon®	E7-8890 v4	
Intel®	Xeon®	E7-8891 v4	
Intel®	Xeon®	E7-8893 v4	
Intel®	Xeon®	E7-8894 v4	
Intel®	Xeon®	Gold 5115	
Intel®	Xeon®	Gold 5118	
Intel®	Xeon®	Gold 5119T	
Intel®	Xeon®	Gold 5120	
Intel®	Xeon®	Gold 5120T	
Intel®	Xeon®	Gold 5122	
Intel®	Xeon®	Gold 5215	
Intel®	Xeon®	Gold 5215L	
Intel®	Xeon®	Gold 5217	
Intel®	Xeon®	Gold 5218	
Intel®	Xeon®	Gold 5218B	
Intel®	Xeon®	Gold 5218N	
Intel®	Xeon®	Gold 5218R	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 5218T	
Intel®	Xeon®	Gold 5220	
Intel®	Xeon®	Gold 5220R	
Intel®	Xeon®	Gold 5220S	
Intel®	Xeon®	Gold 5220T	
Intel®	Xeon®	Gold 5222	
Intel®	Xeon®	Gold 5315Y	
Intel®	Xeon®	Gold 5317	
Intel®	Xeon®	Gold 5318N	
Intel®	Xeon®	Gold 5318S	
Intel®	Xeon®	Gold 5320	
Intel®	Xeon®	Gold 5320T	
Intel®	Xeon®	Gold 6126	
Intel®	Xeon®	Gold 6126F	
Intel®	Xeon®	Gold 6126T	
Intel®	Xeon®	Gold 6128	
Intel®	Xeon®	Gold 6130	
Intel®	Xeon®	Gold 6130F	
Intel®	Xeon®	Gold 6130T	
Intel®	Xeon®	Gold 6132	
Intel®	Xeon®	Gold 6134	
Intel®	Xeon®	Gold 6136	
Intel®	Xeon®	Gold 6138	
Intel®	Xeon®	Gold 6138F	
Intel®	Xeon®	Gold 6138P	
Intel®	Xeon®	Gold 6138T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6140	
Intel®	Xeon®	Gold 6142	
Intel®	Xeon®	Gold 6142F	
Intel®	Xeon®	Gold 6144	
Intel®	Xeon®	Gold 6146	
Intel®	Xeon®	Gold 6148	
Intel®	Xeon®	Gold 6148F	
Intel®	Xeon®	Gold 6150	
Intel®	Xeon®	Gold 6152	
Intel®	Xeon®	Gold 6154	
Intel®	Xeon®	Gold 6208U	
Intel®	Xeon®	Gold 6209U	
Intel®	Xeon®	Gold 6210U	
Intel®	Xeon®	Gold 6212U	
Intel®	Xeon®	Gold 6222V	
Intel®	Xeon®	Gold 6226	
Intel®	Xeon®	Gold 6226R	
Intel®	Xeon®	Gold 6230	
Intel®	Xeon®	Gold 6230N	
Intel®	Xeon®	Gold 6230R	
Intel®	Xeon®	Gold 6230T	
Intel®	Xeon®	Gold 6234	
Intel®	Xeon®	Gold 6238	
Intel®	Xeon®	Gold 6238L	
Intel®	Xeon®	Gold 6238R	
Intel®	Xeon®	Gold 6238T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6240	
Intel®	Xeon®	Gold 6240L	
Intel®	Xeon®	Gold 6240R	
Intel®	Xeon®	Gold 6240Y	
Intel®	Xeon®	Gold 6242	
Intel®	Xeon®	Gold 6242R	
Intel®	Xeon®	Gold 6244	
Intel®	Xeon®	Gold 6246	
Intel®	Xeon®	Gold 6246R	
Intel®	Xeon®	Gold 6248	
Intel®	Xeon®	Gold 6248R	
Intel®	Xeon®	Gold 6250	
Intel®	Xeon®	Gold 6250L	
Intel®	Xeon®	Gold 6252	
Intel®	Xeon®	Gold 6252N	
Intel®	Xeon®	Gold 6254	
Intel®	Xeon®	Gold 6256	
Intel®	Xeon®	Gold 6258R	
Intel®	Xeon®	Gold 6262V	
Intel®	Xeon®	Gold 6312U	
Intel®	Xeon®	Gold 6314U	
Intel®	Xeon®	Gold 6326	
Intel®	Xeon®	Gold 6330	
Intel®	Xeon®	Gold 6330N	
Intel®	Xeon®	Gold 6334	
Intel®	Xeon®	Gold 6336Y	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6338	
Intel®	Xeon®	Gold 6338N	
Intel®	Xeon®	Gold 6338T	
Intel®	Xeon®	Gold 6342	
Intel®	Xeon®	Gold 6346	
Intel®	Xeon®	Gold 6348	
Intel®	Xeon®	Gold 6354	
Intel®	Xeon®	Gold Gold 5318Y	
Intel®	Xeon®	Platinum 8153	
Intel®	Xeon®	Platinum 8156	
Intel®	Xeon®	Platinum 8158	
Intel®	Xeon®	Platinum 8160	
Intel®	Xeon®	Platinum 8160F	
Intel®	Xeon®	Platinum 8160T	
Intel®	Xeon®	Platinum 8164	
Intel®	Xeon®	Platinum 8168	
Intel®	Xeon®	Platinum 8170	
Intel®	Xeon®	Platinum 8176	
Intel®	Xeon®	Platinum 8176F	
Intel®	Xeon®	Platinum 8180	
Intel®	Xeon®	Platinum 8253	
Intel®	Xeon®	Platinum 8256	
Intel®	Xeon®	Platinum 8260	
Intel®	Xeon®	Platinum 8260L	
Intel®	Xeon®	Platinum 8260Y	
Intel®	Xeon®	Platinum 8268	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Platinum 8270	
Intel®	Xeon®	Platinum 8276	
Intel®	Xeon®	Platinum 8276L	
Intel®	Xeon®	Platinum 8280	
Intel®	Xeon®	Platinum 8280L	
Intel®	Xeon®	Platinum 8351N	
Intel®	Xeon®	Platinum 8352M	
Intel®	Xeon®	Platinum 8352S	
Intel®	Xeon®	Platinum 8352V	
Intel®	Xeon®	Platinum 8352Y	
Intel®	Xeon®	Platinum 8358	
Intel®	Xeon®	Platinum 8358P	
Intel®	Xeon®	Platinum 8360Y	
Intel®	Xeon®	Platinum 8362	
Intel®	Xeon®	Platinum 8368	
Intel®	Xeon®	Platinum 8368Q	
Intel®	Xeon®	Platinum 8380	
Intel®	Xeon®	Platinum 9221	
Intel®	Xeon®	Platinum 9222	
Intel®	Xeon®	Platinum 9242	
Intel®	Xeon®	Platinum 9282	
Intel®	Xeon®	Silver 4108	
Intel®	Xeon®	Silver 4109T	
Intel®	Xeon®	Silver 4110	
Intel®	Xeon®	Silver 4112	
Intel®	Xeon®	Silver 4114	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Silver 4114T	
Intel®	Xeon®	Silver 4116	
Intel®	Xeon®	Silver 4116T	
Intel®	Xeon®	Silver 4208	
Intel®	Xeon®	Silver 4209T	
Intel®	Xeon®	Silver 4210	
Intel®	Xeon®	Silver 4210R	
Intel®	Xeon®	Silver 4210T	
Intel®	Xeon®	Silver 4214	
Intel®	Xeon®	Silver 4214R	
Intel®	Xeon®	Silver 4214Y	
Intel®	Xeon®	Silver 4215	
Intel®	Xeon®	Silver 4215R	
Intel®	Xeon®	Silver 4216	
Intel®	Xeon®	Silver 4309Y	
Intel®	Xeon®	Silver 4310	
Intel®	Xeon®	Silver 4310T	
Intel®	Xeon®	Silver 4314	
Intel®	Xeon®	Silver 4316	
Intel®	Xeon®	W-10855M	
Intel®	Xeon®	W-10885M	
Intel®	Xeon®	W-11855M	
Intel®	Xeon®	W-11955M	
Intel®	Xeon®	W-1250	
Intel®	Xeon®	W-1250E	
Intel®	Xeon®	W-1250P	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-1250TE	
Intel®	Xeon®	W-1270	
Intel®	Xeon®	W-1270E	
Intel®	Xeon®	W-1270P	
Intel®	Xeon®	W-1270TE	
Intel®	Xeon®	W-1290	
Intel®	Xeon®	W-1290E	
Intel®	Xeon®	W-1290P	
Intel®	Xeon®	W-1290T	
Intel®	Xeon®	W-1290TE	
Intel®	Xeon®	W-2102	
Intel®	Xeon®	W-2104	
Intel®	Xeon®	W-2123	
Intel®	Xeon®	W-2125	
Intel®	Xeon®	W-2133	
Intel®	Xeon®	W-2135	
Intel®	Xeon®	W-2145	
Intel®	Xeon®	W-2155	
Intel®	Xeon®	W-2175	
Intel®	Xeon®	W-2195	
Intel®	Xeon®	W-2223	
Intel®	Xeon®	W-2225	
Intel®	Xeon®	W-2235	
Intel®	Xeon®	W-2245	
Intel®	Xeon®	W-2255	
Intel®	Xeon®	W-2265	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-2275	
Intel®	Xeon®	W-2295	
Intel®	Xeon®	W-3175X	
Intel®	Xeon®	W-3223	
Intel®	Xeon®	W-3225	
Intel®	Xeon®	W-3235	
Intel®	Xeon®	W-3245	
Intel®	Xeon®	W-3245M	
Intel®	Xeon®	W-3265	
Intel®	Xeon®	W-3265M	
Intel®	Xeon®	W-3275	
Intel®	Xeon®	W-3275M	

Windows 10 21H2 Supported Qualcomm Processors

Article • 12/02/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	Snapdragon 8cx
Qualcomm®	Snapdragon™	Snapdragon 8cx Gen 2

Windows 10 22H2 Supported AMD Processors

Article • 05/23/2023

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	AMD	3015e
AMD	AMD	3020e
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution

Manufacturer	Brand	Model
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G

Manufacturer	Brand	Model
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G
AMD	Athlon™	Gold 3150GE
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e

Manufacturer	Brand	Model
AMD	Athlon™	Silver 3050GE
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7313
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	7401
AMD	EPYC™	7402
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453

Manufacturer	Brand	Model
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7551
AMD	EPYC™	7552
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7713
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7232P
AMD	EPYC™	72F3
AMD	EPYC™	7302P
AMD	EPYC™	7313P
AMD	EPYC™	7351P
AMD	EPYC™	73F3
AMD	EPYC™	7401P
AMD	EPYC™	7402P
AMD	EPYC™	7443P

Manufacturer	Brand	Model
AMD	EPYC™	74F3
AMD	EPYC™	7502P
AMD	EPYC™	7543P
AMD	EPYC™	7551P
AMD	EPYC™	75F3
AMD	EPYC™	7702P
AMD	EPYC™	7713P
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	EPYC™	8024P
AMD	EPYC™	8024PN
AMD	EPYC™	8124P
AMD	EPYC™	8124PN
AMD	EPYC™	8224P
AMD	EPYC™	8224PN
AMD	EPYC™	8324P
AMD	EPYC™	8324PN
AMD	EPYC™	8434P
AMD	EPYC™	8434PN
AMD	EPYC™	8534P
AMD	EPYC™	8534PN
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210

Manufacturer	Brand	Model
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300

Manufacturer	Brand	Model
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B

Manufacturer	Brand	Model
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	3100
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G
AMD	Ryzen™ 3	4300GE
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3	5125C
AMD	Ryzen™ 3	5300G (OEM Only)
AMD	Ryzen™ 3	5300GE (OEM Only)
AMD	Ryzen™ 3	5300U
AMD	Ryzen™ 3	5400U
AMD	Ryzen™ 3	5425C
AMD	Ryzen™ 3	5425U
AMD	Ryzen™ 3	7320U

Manufacturer	Brand	Model
AMD	Ryzen™ 3	7335U
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 3 PRO	5350G
AMD	Ryzen™ 3 PRO	5350GE
AMD	Ryzen™ 3 PRO	5450U
AMD	Ryzen™ 3 PRO	5475U
AMD	Ryzen™ 3 PRO	7330U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	5500
AMD	Ryzen™ 5	5600
AMD	Ryzen™ 5	7600
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600E

Manufacturer	Brand	Model
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G
AMD	Ryzen™ 5	4600GE
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5	5300G
AMD	Ryzen™ 5	5300GE
AMD	Ryzen™ 5	5500U
AMD	Ryzen™ 5	5560U
AMD	Ryzen™ 5	5600G
AMD	Ryzen™ 5	5600GE
AMD	Ryzen™ 5	5600H
AMD	Ryzen™ 5	5600HS
AMD	Ryzen™ 5	5600U

Manufacturer	Brand	Model
AMD	Ryzen™ 5	5600X
AMD	Ryzen™ 5	5625C
AMD	Ryzen™ 5	5625U
AMD	Ryzen™ 5	6600H
AMD	Ryzen™ 5	6600HS
AMD	Ryzen™ 5	6600U
AMD	Ryzen™ 5	7520U
AMD	Ryzen™ 5	7535HS
AMD	Ryzen™ 5	7535U
AMD	Ryzen™ 5	7600X
AMD	Ryzen™ 5	7640HS
AMD	Ryzen™ 5	7645HX
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	5645
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 5 PRO	5650G
AMD	Ryzen™ 5 PRO	5650GE
AMD	Ryzen™ 5 PRO	5650U
AMD	Ryzen™ 5 PRO	5675U
AMD	Ryzen™ 5 PRO	5750G

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	5750GE
AMD	Ryzen™ 5 PRO	6650H
AMD	Ryzen™ 5 PRO	6650HS
AMD	Ryzen™ 5 PRO	6650U
AMD	Ryzen™ 5 PRO	7530U
AMD	Ryzen™ 7	1700
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	7700
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G
AMD	Ryzen™ 7	4700GE
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H

Manufacturer	Brand	Model
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7	5700G
AMD	Ryzen™ 7	5700GE
AMD	Ryzen™ 7	5700U
AMD	Ryzen™ 7	5700X
AMD	Ryzen™ 7	5800 (OEM
AMD	Ryzen™ 7	5800H
AMD	Ryzen™ 7	5800HS
AMD	Ryzen™ 7	5800U
AMD	Ryzen™ 7	5800X
AMD	Ryzen™ 7	5800X3D
AMD	Ryzen™ 7	5825C
AMD	Ryzen™ 7	5825U
AMD	Ryzen™ 7	6800H
AMD	Ryzen™ 7	6800HS
AMD	Ryzen™ 7	6800U
AMD	Ryzen™ 7	7700X
AMD	Ryzen™ 7	7730U
AMD	Ryzen™ 7	7735HS
AMD	Ryzen™ 7	7735U
AMD	Ryzen™ 7	7736U
AMD	Ryzen™ 7	7800X3D
AMD	Ryzen™ 7	7840HS
AMD	Ryzen™ 7 PRO	5845
AMD	Ryzen™ 7 PRO	4750G

Manufacturer	Brand	Model
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 7 PRO	5850U
AMD	Ryzen™ 7 PRO	5875U
AMD	Ryzen™ 7 PRO	6850H
AMD	Ryzen™ 7 PRO	6850HS
AMD	Ryzen™ 7 PRO	6850U
AMD	Ryzen™ 7 PRO	6860Z
AMD	Ryzen™ 9	5900
AMD	Ryzen™ 9	7900
AMD	Ryzen™ 9	3900 Processor
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9	5900 (OEM
AMD	Ryzen™ 9	5900HS
AMD	Ryzen™ 9	5900HX
AMD	Ryzen™ 9	5900X
AMD	Ryzen™ 9	5950X
AMD	Ryzen™ 9	5980HS
AMD	Ryzen™ 9	5980HX
AMD	Ryzen™ 9	6900HS
AMD	Ryzen™ 9	6900HX
AMD	Ryzen™ 9	6980HS

Manufacturer	Brand	Model
AMD	Ryzen™ 9	6980HX
AMD	Ryzen™ 9	7900X
AMD	Ryzen™ 9	7900X3D
AMD	Ryzen™ 9	7940HS
AMD	Ryzen™ 9	7950X
AMD	Ryzen™ 9	7950X3D
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ 9 PRO	5945
AMD	Ryzen™ 9 PRO	6950H
AMD	Ryzen™ 9 PRO	6950HS
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Manufacturer	Brand	Model
AMD	Ryzen™ Threadripper™ PRO	5945WX
AMD	Ryzen™ Threadripper™ PRO	5955WX
AMD	Ryzen™ Threadripper™ PRO	5965WX
AMD	Ryzen™ Threadripper™ PRO	5975WX
AMD	Ryzen™ Threadripper™ PRO	5995WX

Windows 10 22H2 Supported Intel Processors

Article • 05/23/2023

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Intel®	Atom®	E3805	IoT Enterprise Only
Intel®	Atom®	E3815	IoT Enterprise Only
Intel®	Atom®	E3825	IoT Enterprise Only
Intel®	Atom®	E3826	IoT Enterprise Only
Intel®	Atom®	E3827	IoT Enterprise Only
Intel®	Atom®	E3845	IoT Enterprise Only
Intel®	Atom®	x5-E3930	IoT Enterprise Only
Intel®	Atom®	x5-E3940	IoT Enterprise Only
Intel®	Atom®	x5-E3950	IoT Enterprise Only
Intel®	Atom®	x5-E8000	IoT Enterprise Only
Intel®	Atom®	x6200FE	
Intel®	Atom®	x6211E	
Intel®	Atom®	x6212RE	
Intel®	Atom®	x6413E	
Intel®	Atom®	x6414RE	
Intel®	Atom®	x6425E	
Intel®	Atom®	x6425RE	
Intel®	Atom®	x6427FE	
Intel®	Celeron®	6305	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	7300	
Intel®	Celeron®	7305	
Intel®	Celeron®	3205U	
Intel®	Celeron®	3215U	
Intel®	Celeron®	3755U	
Intel®	Celeron®	3765U	
Intel®	Celeron®	3865U	
Intel®	Celeron®	3867U	
Intel®	Celeron®	3965U	
Intel®	Celeron®	3965Y	
Intel®	Celeron®	4205U	
Intel®	Celeron®	4305U	
Intel®	Celeron®	4305UE	
Intel®	Celeron®	5205U	
Intel®	Celeron®	5305U	
Intel®	Celeron®	6305E	
Intel®	Celeron®	6600HE	
Intel®	Celeron®	7305E	IoT Enterprise Only
Intel®	Celeron®	7305L	
Intel®	Celeron®	G3930	
Intel®	Celeron®	G3930E	
Intel®	Celeron®	G3930T	
Intel®	Celeron®	G3930TE	
Intel®	Celeron®	G3950	
Intel®	Celeron®	G4900	
Intel®	Celeron®	G4900T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	G4920	
Intel®	Celeron®	G4930	
Intel®	Celeron®	G4930E	
Intel®	Celeron®	G4930T	
Intel®	Celeron®	G4932E	
Intel®	Celeron®	G4950	
Intel®	Celeron®	G5900	
Intel®	Celeron®	G5900E	
Intel®	Celeron®	G5900T	
Intel®	Celeron®	G5900TE	
Intel®	Celeron®	G5905	
Intel®	Celeron®	G5905T	
Intel®	Celeron®	G5920	
Intel®	Celeron®	G5925	
Intel®	Celeron®	G6900	
Intel®	Celeron®	G6900E	IoT Enterprise Only
Intel®	Celeron®	G6900T	
Intel®	Celeron®	G6900TE	IoT Enterprise Only
Intel®	Celeron®	J1900	IoT Enterprise Only
Intel®	Celeron®	J3355	IoT Enterprise Only
Intel®	Celeron®	J3355E	IoT Enterprise Only
Intel®	Celeron®	J3455	IoT Enterprise Only
Intel®	Celeron®	J3455E	IoT Enterprise Only
Intel®	Celeron®	J4005	
Intel®	Celeron®	J4025	
Intel®	Celeron®	J4105	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	J4115	
Intel®	Celeron®	J4125	
Intel®	Celeron®	J6412	
Intel®	Celeron®	J6413	
Intel®	Celeron®	N4000	
Intel®	Celeron®	N4020	
Intel®	Celeron®	N4100	
Intel®	Celeron®	N4120	
Intel®	Celeron®	N4500	
Intel®	Celeron®	N4505	
Intel®	Celeron®	N5100	
Intel®	Celeron®	N5105	
Intel®	Celeron®	N6210	
Intel®	Celeron®	N6211	
Intel®	Core™	i3-1000G1	
Intel®	Core™	i3-1000G4	
Intel®	Core™	i3-1005G1	
Intel®	Core™	i3-10100	
Intel®	Core™	i3-10100E	
Intel®	Core™	i3-10100F	
Intel®	Core™	i3-10100T	
Intel®	Core™	i3-10100TE	
Intel®	Core™	i3-10100Y	
Intel®	Core™	i3-10105	
Intel®	Core™	i3-10105F	
Intel®	Core™	i3-10105T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-10110U	
Intel®	Core™	i3-10110Y	
Intel®	Core™	i3-10300	
Intel®	Core™	i3-10300T	
Intel®	Core™	i3-10305	
Intel®	Core™	i3-10305T	
Intel®	Core™	i3-10320	
Intel®	Core™	i3-10325	
Intel®	Core™	i3-11100HE	
Intel®	Core™	i3-1110G4	
Intel®	Core™	i3-1115G4	
Intel®	Core™	i3-1115G4E	
Intel®	Core™	i3-1115GRE	
Intel®	Core™	i3-1120G4	
Intel®	Core™	i3-1125G4	
Intel®	Core™	i3-12100	
Intel®	Core™	i3-12100E	IoT Enterprise Only
Intel®	Core™	i3-12100F	
Intel®	Core™	i3-12100T	
Intel®	Core™	i3-12100TE	IoT Enterprise Only
Intel®	Core™	i3-1210U	
Intel®	Core™	i3-1215U	
Intel®	Core™	i3-1215UE	IoT Enterprise Only
Intel®	Core™	i3-1215UL	
Intel®	Core™	i3-1220P	
Intel®	Core™	i3-1220PE	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-12300	
Intel®	Core™	i3-12300HE	IoT Enterprise Only
Intel®	Core™	i3-12300HL	
Intel®	Core™	i3-12300T	
Intel®	Core™	i3-1305U	
Intel®	Core™	i3-13100	
Intel®	Core™	i3-13100E	
Intel®	Core™	i3-13100F	
Intel®	Core™	i3-13100T	
Intel®	Core™	i3-13100TE	
Intel®	Core™	i3-1315U	
Intel®	Core™	i3-1315UE	
Intel®	Core™	i3-1320PE	
Intel®	Core™	i3-13300HE	
Intel®	Core™	i3-5005U	
Intel®	Core™	i3-5010U	
Intel®	Core™	i3-5015U	
Intel®	Core™	i3-5020U	
Intel®	Core™	i3-5157U	
Intel®	Core™	i3-7020U	
Intel®	Core™	i3-7100	
Intel®	Core™	i3-7100E	
Intel®	Core™	i3-7100H	
Intel®	Core™	i3-7100T	
Intel®	Core™	i3-7100U	
Intel®	Core™	i3-7101E	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-7101TE	
Intel®	Core™	i3-7102E	
Intel®	Core™	i3-7130U	
Intel®	Core™	i3-7167U	
Intel®	Core™	i3-7300	
Intel®	Core™	i3-7300T	
Intel®	Core™	i3-7320	
Intel®	Core™	i3-7350K	
Intel®	Core™	i3-8100	
Intel®	Core™	i3-8100B	
Intel®	Core™	i3-8100H	
Intel®	Core™	i3-8100T	
Intel®	Core™	i3-8109U	
Intel®	Core™	i3-8130U	
Intel®	Core™	i3-8140U	
Intel®	Core™	i3-8145U	
Intel®	Core™	i3-8145UE	
Intel®	Core™	i3-8300	
Intel®	Core™	i3-8300T	
Intel®	Core™	i3-8350K	
Intel®	Core™	i3-9100	
Intel®	Core™	i3-9100E	
Intel®	Core™	i3-9100F	
Intel®	Core™	i3-9100HL	
Intel®	Core™	i3-9100T	
Intel®	Core™	i3-9100TE	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-9300	
Intel®	Core™	i3-9300T	
Intel®	Core™	i3-9320	
Intel®	Core™	i3-9350K	
Intel®	Core™	i3-9350KF	
Intel®	Core™	i3-L13G4	
Intel®	Core™	i5-10200H	
Intel®	Core™	i5-10210U	
Intel®	Core™	i5-10210Y	
Intel®	Core™	i5-10300H	
Intel®	Core™	i5-1030G4	
Intel®	Core™	i5-1030G7	
Intel®	Core™	i5-10310U	
Intel®	Core™	i5-10310Y	
Intel®	Core™	i5-1035G1	
Intel®	Core™	i5-1035G4	
Intel®	Core™	i5-1035G7	
Intel®	Core™	i5-1038NG7	
Intel®	Core™	i5-10400	
Intel®	Core™	i5-10400F	
Intel®	Core™	i5-10400H	
Intel®	Core™	i5-10400T	
Intel®	Core™	i5-10500	
Intel®	Core™	i5-10500E	
Intel®	Core™	i5-10500H	
Intel®	Core™	i5-10500T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-10500TE	
Intel®	Core™	i5-10505	
Intel®	Core™	i5-10600	
Intel®	Core™	i5-10600K	
Intel®	Core™	i5-10600KF	
Intel®	Core™	i5-10600T	
Intel®	Core™	i5-11260H	
Intel®	Core™	i5-11300H	
Intel®	Core™	i5-1130G7	
Intel®	Core™	i5-11320H	
Intel®	Core™	i5-1135G7	
Intel®	Core™	i5-11400	
Intel®	Core™	i5-11400F	
Intel®	Core™	i5-11400H	
Intel®	Core™	i5-11400T	
Intel®	Core™	i5-1140G7	
Intel®	Core™	i5-1145G7	
Intel®	Core™	i5-1145G7E	
Intel®	Core™	i5-1145GRE	
Intel®	Core™	i5-11500	
Intel®	Core™	i5-11500H	
Intel®	Core™	i5-11500HE	
Intel®	Core™	i5-11500T	
Intel®	Core™	i5-1155G7	
Intel®	Core™	i5-11600	
Intel®	Core™	i5-11600K	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-11600KF	
Intel®	Core™	i5-11600T	
Intel®	Core™	i5-1230U	
Intel®	Core™	i5-1235U	
Intel®	Core™	i5-1235UL	
Intel®	Core™	i5-12400	
Intel®	Core™	i5-12400F	
Intel®	Core™	i5-12400T	
Intel®	Core™	i5-1240P	
Intel®	Core™	i5-1240U	
Intel®	Core™	i5-12450H	
Intel®	Core™	i5-12450HX	
Intel®	Core™	i5-1245U	
Intel®	Core™	i5-1245UE	IoT Enterprise Only
Intel®	Core™	i5-1245UL	
Intel®	Core™	i5-12500	
Intel®	Core™	i5-12500E	IoT Enterprise Only
Intel®	Core™	i5-12500H	
Intel®	Core™	i5-12500HL	
Intel®	Core™	i5-12500T	
Intel®	Core™	i5-12500TE	IoT Enterprise Only
Intel®	Core™	i5-1250P	
Intel®	Core™	i5-1250PE	IoT Enterprise Only
Intel®	Core™	i5-12600	
Intel®	Core™	i5-12600H	
Intel®	Core™	i5-12600HE	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-12600HL	
Intel®	Core™	i5-12600HX	
Intel®	Core™	i5-12600K	
Intel®	Core™	i5-12600KF	
Intel®	Core™	i5-12600T	
Intel®	Core™	i5-1334U	
Intel®	Core™	i5-1335U	
Intel®	Core™	i5-1335UE	
Intel®	Core™	i5-13400	
Intel®	Core™	i5-13400E	
Intel®	Core™	i5-13400F	
Intel®	Core™	i5-13400T	
Intel®	Core™	i5-1340P	
Intel®	Core™	i5-1340PE	
Intel®	Core™	i5-13420H	
Intel®	Core™	i5-13450HX	
Intel®	Core™	i5-1345U	
Intel®	Core™	i5-1345UE	
Intel®	Core™	i5-13490F	
Intel®	Core™	i5-13500	
Intel®	Core™	i5-13500E	
Intel®	Core™	i5-13500H	
Intel®	Core™	i5-13500HX	
Intel®	Core™	i5-13500T	
Intel®	Core™	i5-13500TE	
Intel®	Core™	i5-13505H	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-1350P	
Intel®	Core™	i5-1350PE	
Intel®	Core™	i5-13600	
Intel®	Core™	i5-13600H	
Intel®	Core™	i5-13600HE	
Intel®	Core™	i5-13600HX	
Intel®	Core™	i5-13600K	
Intel®	Core™	i5-13600KF	
Intel®	Core™	i5-13600T	
Intel®	Core™	i5-5200U	
Intel®	Core™	i5-5250U	
Intel®	Core™	i5-5257U	
Intel®	Core™	i5-5287U	
Intel®	Core™	i5-5300U	
Intel®	Core™	i5-5350H	
Intel®	Core™	i5-5350U	
Intel®	Core™	i5-5575R	
Intel®	Core™	i5-5675C	
Intel®	Core™	i5-5675R	
Intel®	Core™	i5-7200U	
Intel®	Core™	i5-7260U	
Intel®	Core™	i5-7267U	
Intel®	Core™	i5-7287U	
Intel®	Core™	i5-7300HQ	
Intel®	Core™	i5-7300U	
Intel®	Core™	i5-7360U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-7400	
Intel®	Core™	i5-7400T	
Intel®	Core™	i5-7440EQ	
Intel®	Core™	i5-7440HQ	
Intel®	Core™	i5-7442EQ	
Intel®	Core™	i5-7500	
Intel®	Core™	i5-7500T	
Intel®	Core™	i5-7600	
Intel®	Core™	i5-7600K	
Intel®	Core™	i5-7600T	
Intel®	Core™	i5-7640X	
Intel®	Core™	i5-7Y54	
Intel®	Core™	i5-7Y57	
Intel®	Core™	i5-8200Y	
Intel®	Core™	i5-8210Y	
Intel®	Core™	i5-8250U	
Intel®	Core™	i5-8257U	
Intel®	Core™	i5-8259U	
Intel®	Core™	i5-8260U	
Intel®	Core™	i5-8265U	
Intel®	Core™	i5-8269U	
Intel®	Core™	i5-8279U	
Intel®	Core™	i5-8300H	
Intel®	Core™	i5-8305G	
Intel®	Core™	i5-8310Y	
Intel®	Core™	i5-8350U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-8365U	
Intel®	Core™	i5-8365UE	
Intel®	Core™	i5-8400	
Intel®	Core™	i5-8400B	
Intel®	Core™	i5-8400H	
Intel®	Core™	i5-8400T	
Intel®	Core™	i5-8500	
Intel®	Core™	i5-8500B	
Intel®	Core™	i5-8500T	
Intel®	Core™	i5-8600	
Intel®	Core™	i5-8600K	
Intel®	Core™	i5-8600T	
Intel®	Core™	i5-9300H	
Intel®	Core™	i5-9300HF	
Intel®	Core™	i5-9400	
Intel®	Core™	i5-9400F	
Intel®	Core™	i5-9400H	
Intel®	Core™	i5-9400T	
Intel®	Core™	i5-9500	
Intel®	Core™	i5-9500E	
Intel®	Core™	i5-9500F	
Intel®	Core™	i5-9500T	
Intel®	Core™	i5-9500TE	
Intel®	Core™	i5-9600	
Intel®	Core™	i5-9600K	
Intel®	Core™	i5-9600KF	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-9600T	
Intel®	Core™	i5-L16G7	
Intel®	Core™	i7-10510U	
Intel®	Core™	i7-10510Y	
Intel®	Core™	i7-1060G7	
Intel®	Core™	i7-10610U	
Intel®	Core™	i7-1065G7	
Intel®	Core™	i7-1068NG7	
Intel®	Core™	i7-10700	
Intel®	Core™	i7-10700E	
Intel®	Core™	i7-10700F	
Intel®	Core™	i7-10700K	
Intel®	Core™	i7-10700KF	
Intel®	Core™	i7-10700T	
Intel®	Core™	i7-10700TE	
Intel®	Core™	i7-10710U	
Intel®	Core™	i7-10750H	
Intel®	Core™	i7-10810U	
Intel®	Core™	i7-10850H	
Intel®	Core™	i7-10870H	
Intel®	Core™	i7-10875H	
Intel®	Core™	i7-11370H	
Intel®	Core™	i7-11375H	
Intel®	Core™	i7-11390H	
Intel®	Core™	i7-11600H	
Intel®	Core™	i7-1160G7	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-11700	
Intel®	Core™	i7-11700F	
Intel®	Core™	i7-11700K	
Intel®	Core™	i7-11700KF	
Intel®	Core™	i7-11700T	
Intel®	Core™	i7-11800H	
Intel®	Core™	i7-1180G7	
Intel®	Core™	i7-11850H	
Intel®	Core™	i7-11850HE	
Intel®	Core™	i7-1185G7	
Intel®	Core™	i7-1185G7E	
Intel®	Core™	i7-1185GRE	
Intel®	Core™	i7-1195G7	
Intel®	Core™	i7-1250U	
Intel®	Core™	i7-1255U	
Intel®	Core™	i7-1255UL	
Intel®	Core™	i7-1260P	
Intel®	Core™	i7-1260U	
Intel®	Core™	i7-12650H	
Intel®	Core™	i7-12650HX	
Intel®	Core™	i7-1265U	
Intel®	Core™	i7-1265UE	IoT Enterprise Only
Intel®	Core™	i7-1265UL	
Intel®	Core™	i7-12700	
Intel®	Core™	i7-12700E	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-12700F	
Intel®	Core™	i7-12700H	
Intel®	Core™	i7-12700HL	
Intel®	Core™	i7-12700K	
Intel®	Core™	i7-12700KF	
Intel®	Core™	i7-12700T	
Intel®	Core™	i7-12700TE	IoT Enterprise Only
Intel®	Core™	i7-1270P	
Intel®	Core™	i7-1270PE	IoT Enterprise Only
Intel®	Core™	i7-12800H	
Intel®	Core™	i7-12800HE	IoT Enterprise Only
Intel®	Core™	i7-12800HL	
Intel®	Core™	i7-12800HX	
Intel®	Core™	i7-1280P	
Intel®	Core™	i7-12850HX	
Intel®	Core™	i7-1355U	
Intel®	Core™	i7-1360P	
Intel®	Core™	i7-13620H	
Intel®	Core™	i7-13650HX	
Intel®	Core™	i7-1365U	
Intel®	Core™	i7-1365UE	
Intel®	Core™	i7-13700	
Intel®	Core™	i7-13700E	
Intel®	Core™	i7-13700F	
Intel®	Core™	i7-13700H	
Intel®	Core™	i7-13700HX	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-13700K	
Intel®	Core™	i7-13700KF	
Intel®	Core™	i7-13700T	
Intel®	Core™	i7-13700TE	
Intel®	Core™	i7-13705H	
Intel®	Core™	i7-1370P	
Intel®	Core™	i7-1370PE	
Intel®	Core™	i7-13790F	
Intel®	Core™	i7-13800H	
Intel®	Core™	i7-13800HE	
Intel®	Core™	i7-13850HX	
Intel®	Core™	i7-5500U	
Intel®	Core™	i7-5550U	
Intel®	Core™	i7-5557U	
Intel®	Core™	i7-5600U	
Intel®	Core™	i7-5650U	
Intel®	Core™	i7-5700EQ	
Intel®	Core™	i7-5700HQ	
Intel®	Core™	i7-5750HQ	
Intel®	Core™	i7-5775C	
Intel®	Core™	i7-5775R	
Intel®	Core™	i7-5850EQ	
Intel®	Core™	i7-5850HQ	
Intel®	Core™	i7-5950HQ	
Intel®	Core™	i7-7560U	
Intel®	Core™	i7-7567U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-7600U	
Intel®	Core™	i7-7660U	
Intel®	Core™	i7-7700	
Intel®	Core™	i7-7700HQ	
Intel®	Core™	i7-7700K	
Intel®	Core™	i7-7700T	
Intel®	Core™	i7-7740X	
Intel®	Core™	i7-7800X	
Intel®	Core™	i7-7820EQ	
Intel®	Core™	i7-7820HK	
Intel®	Core™	i7-7820HQ	
Intel®	Core™	i7-7820X	
Intel®	Core™	i7-7920HQ	
Intel®	Core™	i7-7Y75	
Intel®	Core™	i7-8086K	
Intel®	Core™	i7-8500Y	
Intel®	Core™	i7-8550U	
Intel®	Core™	i7-8557U	
Intel®	Core™	i7-8559U	
Intel®	Core™	i7-8569U	
Intel®	Core™	i7-8650U	
Intel®	Core™	i7-8665U	
Intel®	Core™	i7-8665UE	
Intel®	Core™	i7-8700	
Intel®	Core™	i7-8700B	
Intel®	Core™	i7-8700K	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-8700T	
Intel®	Core™	i7-8705G	
Intel®	Core™	i7-8706G	
Intel®	Core™	i7-8709G	
Intel®	Core™	i7-8750H	
Intel®	Core™	i7-8809G	
Intel®	Core™	i7-8850H	
Intel®	Core™	i7-9700	
Intel®	Core™	i7-9700E	
Intel®	Core™	i7-9700F	
Intel®	Core™	i7-9700K	
Intel®	Core™	i7-9700KF	
Intel®	Core™	i7-9700T	
Intel®	Core™	i7-9700TE	
Intel®	Core™	i7-9750H	
Intel®	Core™	i7-9750HF	
Intel®	Core™	i7-9800X	
Intel®	Core™	i7-9850H	
Intel®	Core™	i7-9850HE	
Intel®	Core™	i7-9850HL	
Intel®	Core™	i9-10850K	
Intel®	Core™	i9-10885H	
Intel®	Core™	i9-10900	
Intel®	Core™	i9-10900E	
Intel®	Core™	i9-10900F	
Intel®	Core™	i9-10900K	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-10900KF	
Intel®	Core™	i9-10900T	
Intel®	Core™	i9-10900TE	
Intel®	Core™	i9-10900X	
Intel®	Core™	i9-10920X	
Intel®	Core™	i9-10940X	
Intel®	Core™	i9-10980HK	
Intel®	Core™	i9-10980XE	
Intel®	Core™	i9-11900	
Intel®	Core™	i9-11900F	
Intel®	Core™	i9-11900H	
Intel®	Core™	i9-11900K	
Intel®	Core™	i9-11900KF	
Intel®	Core™	i9-11900T	
Intel®	Core™	i9-11950H	
Intel®	Core™	i9-11980HK	
Intel®	Core™	i9-12900	
Intel®	Core™	i9-12900E	IoT Enterprise Only
Intel®	Core™	i9-12900F	
Intel®	Core™	i9-12900H	
Intel®	Core™	i9-12900HK	
Intel®	Core™	i9-12900HX	
Intel®	Core™	i9-12900K	
Intel®	Core™	i9-12900KF	
Intel®	Core™	i9-12900KS	
Intel®	Core™	i9-12900T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-12900TE	IoT Enterprise Only
Intel®	Core™	i9-12950HX	
Intel®	Core™	i9-13900	
Intel®	Core™	i9-13900E	
Intel®	Core™	i9-13900F	
Intel®	Core™	i9-13900H	
Intel®	Core™	i9-13900HK	
Intel®	Core™	i9-13900HX	
Intel®	Core™	i9-13900K	
Intel®	Core™	i9-13900KF	
Intel®	Core™	i9-13900KS	
Intel®	Core™	i9-13900T	
Intel®	Core™	i9-13900TE	
Intel®	Core™	i9-13905H	
Intel®	Core™	i9-13950HX	
Intel®	Core™	i9-13980HX	
Intel®	Core™	i9-7900X	
Intel®	Core™	i9-7920X	
Intel®	Core™	i9-7940X	
Intel®	Core™	i9-7960X	
Intel®	Core™	i9-7980XE	
Intel®	Core™	i9-8950HK	
Intel®	Core™	i9-9820X	
Intel®	Core™	i9-9880H	
Intel®	Core™	i9-9900	
Intel®	Core™	i9-9900K	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-9900KF	
Intel®	Core™	i9-9900KS	
Intel®	Core™	i9-9900T	
Intel®	Core™	i9-9900X	
Intel®	Core™	i9-9920X	
Intel®	Core™	i9-9940X	
Intel®	Core™	i9-9960X	
Intel®	Core™	i9-9980HK	
Intel®	Core™	i9-9980XE	
Intel®	Core™	m3-7Y30	
Intel®	Core™	m3-7Y32	
Intel®	Core™	m3-8100Y	
Intel®	Core™	M-5Y10	
Intel®	Core™	M-5Y10a	
Intel®	Core™	M-5Y10c	
Intel®	Core™	M-5Y31	
Intel®	Core™	M-5Y51	
Intel®	Core™	M-5Y70	
Intel®	Core™	M-5Y71	
Intel®	Pentium®	6805	
Intel®	Pentium®	3805U	
Intel®	Pentium®	3825U	
Intel®	Pentium®	D1507	
Intel®	Pentium®	D1508	
Intel®	Pentium®	D1509	
Intel®	Pentium®	D1517	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	D1519	
Intel®	Pentium®	G4560	
Intel®	Pentium®	G4560T	
Intel®	Pentium®	G4600	
Intel®	Pentium®	G4600T	
Intel®	Pentium®	G4620	
Intel®	Pentium®	Gold 4410Y	
Intel®	Pentium®	Gold 4415U	
Intel®	Pentium®	Gold 4415Y	
Intel®	Pentium®	Gold 4417U	
Intel®	Pentium®	Gold 4425Y	
Intel®	Pentium®	Gold 5405U	
Intel®	Pentium®	Gold 6405U	
Intel®	Pentium®	Gold 6500Y	
Intel®	Pentium®	Gold 7505	
Intel®	Pentium®	Gold 8500	
Intel®	Pentium®	Gold 8505	
Intel®	Pentium®	Gold G5400	
Intel®	Pentium®	Gold G5400T	
Intel®	Pentium®	Gold G5420	
Intel®	Pentium®	Gold G5420T	
Intel®	Pentium®	Gold G5500	
Intel®	Pentium®	Gold G5500T	
Intel®	Pentium®	Gold G5600	
Intel®	Pentium®	Gold G5600T	
Intel®	Pentium®	Gold G5620	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	Gold G6400	
Intel®	Pentium®	Gold G6400E	
Intel®	Pentium®	Gold G6400T	
Intel®	Pentium®	Gold G6400TE	
Intel®	Pentium®	Gold G6405	
Intel®	Pentium®	Gold G6405T	
Intel®	Pentium®	Gold G6500	
Intel®	Pentium®	Gold G6500T	
Intel®	Pentium®	Gold G6505	
Intel®	Pentium®	Gold G6505T	
Intel®	Pentium®	Gold G6600	
Intel®	Pentium®	Gold G6605	
Intel®	Pentium®	Gold G7400	
Intel®	Pentium®	Gold G7400E	IoT Enterprise Only
Intel®	Pentium®	Gold G7400T	
Intel®	Pentium®	Gold G7400TE	IoT Enterprise Only
Intel®	Pentium®	J6426	
Intel®	Pentium®	N3710	IoT Enterprise Only
Intel®	Pentium®	N4200	IoT Enterprise Only
Intel®	Pentium®	N6415	
Intel®	Pentium®	Silver J5005	
Intel®	Pentium®	Silver J5040	
Intel®	Pentium®	Silver N5000	
Intel®	Pentium®	Silver N5030	
Intel®	Pentium®	Silver N6000	
Intel®	Pentium®	Silver N6005	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Processor	U300	
Intel®	Processor	U300E	
Intel®	Xeon®	Bronze 3104	
Intel®	Xeon®	Bronze 3106	
Intel®	Xeon®	Bronze 3204	
Intel®	Xeon®	Bronze 3206R	
Intel®	Xeon®	D-1513N	
Intel®	Xeon®	D-1518	
Intel®	Xeon®	D-1520	
Intel®	Xeon®	D-1521	
Intel®	Xeon®	D-1523N	
Intel®	Xeon®	D-1527	
Intel®	Xeon®	D-1528	
Intel®	Xeon®	D-1529	
Intel®	Xeon®	D-1531	
Intel®	Xeon®	D-1533N	
Intel®	Xeon®	D-1537	
Intel®	Xeon®	D-1539	
Intel®	Xeon®	D-1540	
Intel®	Xeon®	D-1541	
Intel®	Xeon®	D-1543N	
Intel®	Xeon®	D-1548	
Intel®	Xeon®	D-1553N	
Intel®	Xeon®	D-1557	
Intel®	Xeon®	D-1559	
Intel®	Xeon®	D-1567	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	D-1571	
Intel®	Xeon®	D-1577	
Intel®	Xeon®	D-2123IT	
Intel®	Xeon®	D-2141I	
Intel®	Xeon®	D-2142IT	
Intel®	Xeon®	D-2143IT	
Intel®	Xeon®	D-2145NT	
Intel®	Xeon®	D-2146NT	
Intel®	Xeon®	D-2161I	
Intel®	Xeon®	D-2163IT	
Intel®	Xeon®	D-2166NT	
Intel®	Xeon®	D-2173IT	
Intel®	Xeon®	D-2177NT	
Intel®	Xeon®	D-2183IT	
Intel®	Xeon®	D-2187NT	
Intel®	Xeon®	E-2124	
Intel®	Xeon®	E-2124G	
Intel®	Xeon®	E-2126G	
Intel®	Xeon®	E-2134	
Intel®	Xeon®	E-2136	
Intel®	Xeon®	E-2144G	
Intel®	Xeon®	E-2146G	
Intel®	Xeon®	E-2174G	
Intel®	Xeon®	E-2176G	
Intel®	Xeon®	E-2176M	
Intel®	Xeon®	E-2186G	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2186M	
Intel®	Xeon®	E-2224	
Intel®	Xeon®	E-2224G	
Intel®	Xeon®	E-2226G	
Intel®	Xeon®	E-2226GE	
Intel®	Xeon®	E-2234	
Intel®	Xeon®	E-2236	
Intel®	Xeon®	E-2244G	
Intel®	Xeon®	E-2246G	
Intel®	Xeon®	E-2254ME	
Intel®	Xeon®	E-2254ML	
Intel®	Xeon®	E-2274G	
Intel®	Xeon®	E-2276G	
Intel®	Xeon®	E-2276M	
Intel®	Xeon®	E-2276ME	
Intel®	Xeon®	E-2276ML	
Intel®	Xeon®	E-2278G	
Intel®	Xeon®	E-2278GE	
Intel®	Xeon®	E-2278GEL	
Intel®	Xeon®	E-2286G	
Intel®	Xeon®	E-2286M	
Intel®	Xeon®	E-2288G	
Intel®	Xeon®	E-2314	
Intel®	Xeon®	E-2324G	
Intel®	Xeon®	E-2334	
Intel®	Xeon®	E-2336	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2356G	
Intel®	Xeon®	E-2374G	
Intel®	Xeon®	E-2378	
Intel®	Xeon®	E-2378G	
Intel®	Xeon®	E-2386G	
Intel®	Xeon®	E-2388G	
Intel®	Xeon®	E3-1220 v6	
Intel®	Xeon®	E3-1225 v6	
Intel®	Xeon®	E3-1230 v6	
Intel®	Xeon®	E3-1240 v6	
Intel®	Xeon®	E3-1245 v6	
Intel®	Xeon®	E3-1258L v4	
Intel®	Xeon®	E3-1265L v4	
Intel®	Xeon®	E3-1270 v6	
Intel®	Xeon®	E3-1275 v6	
Intel®	Xeon®	E3-1278L v4	
Intel®	Xeon®	E3-1280 v6	
Intel®	Xeon®	E3-1285 v4	
Intel®	Xeon®	E3-1285 v6	
Intel®	Xeon®	E3-1285L v4	
Intel®	Xeon®	E3-1501L v6	
Intel®	Xeon®	E3-1501M v6	
Intel®	Xeon®	E3-1505L v6	
Intel®	Xeon®	E3-1505M v6	
Intel®	Xeon®	E3-1535M v6	
Intel®	Xeon®	E5-1620 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-1630 v4	
Intel®	Xeon®	E5-1650 v4	
Intel®	Xeon®	E5-1660 v4	
Intel®	Xeon®	E5-1680 v4	
Intel®	Xeon®	E5-2603 v4	
Intel®	Xeon®	E5-2608L v4	
Intel®	Xeon®	E5-2609 v4	
Intel®	Xeon®	E5-2618L v4	
Intel®	Xeon®	E5-2620 v4	
Intel®	Xeon®	E5-2623 v4	
Intel®	Xeon®	E5-2628L v4	
Intel®	Xeon®	E5-2630 v4	
Intel®	Xeon®	E5-2630L v4	
Intel®	Xeon®	E5-2637 v4	
Intel®	Xeon®	E5-2640 v4	
Intel®	Xeon®	E5-2643 v4	
Intel®	Xeon®	E5-2648L v4	
Intel®	Xeon®	E5-2650 v4	
Intel®	Xeon®	E5-2650L v4	
Intel®	Xeon®	E5-2658 v4	
Intel®	Xeon®	E5-2660 v4	
Intel®	Xeon®	E5-2667 v4	
Intel®	Xeon®	E5-2680 v4	
Intel®	Xeon®	E5-2683 v4	
Intel®	Xeon®	E5-2687W v4	
Intel®	Xeon®	E5-2690 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-2695 v4	
Intel®	Xeon®	E5-2697 v4	
Intel®	Xeon®	E5-2697A v4	
Intel®	Xeon®	E5-2698 v4	
Intel®	Xeon®	E5-2699 v4	
Intel®	Xeon®	E5-2699A v4	
Intel®	Xeon®	E5-2699R v4	
Intel®	Xeon®	E5-4610 v4	
Intel®	Xeon®	E5-4620 v4	
Intel®	Xeon®	E5-4627 v4	
Intel®	Xeon®	E5-4628L v4	
Intel®	Xeon®	E5-4640 v4	
Intel®	Xeon®	E5-4650 v4	
Intel®	Xeon®	E5-4655 v4	
Intel®	Xeon®	E5-4660 v4	
Intel®	Xeon®	E5-4667 v4	
Intel®	Xeon®	E5-4669 v4	
Intel®	Xeon®	E7-4809 v4	
Intel®	Xeon®	E7-4820 v4	
Intel®	Xeon®	E7-4830 v4	
Intel®	Xeon®	E7-4850 v4	
Intel®	Xeon®	E7-8860 v4	
Intel®	Xeon®	E7-8867 v4	
Intel®	Xeon®	E7-8870 v4	
Intel®	Xeon®	E7-8880 v4	
Intel®	Xeon®	E7-8890 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E7-8891 v4	
Intel®	Xeon®	E7-8893 v4	
Intel®	Xeon®	E7-8894 v4	
Intel®	Xeon®	Gold 5115	
Intel®	Xeon®	Gold 5118	
Intel®	Xeon®	Gold 5119T	
Intel®	Xeon®	Gold 5120	
Intel®	Xeon®	Gold 5120T	
Intel®	Xeon®	Gold 5122	
Intel®	Xeon®	Gold 5215	
Intel®	Xeon®	Gold 5215L	
Intel®	Xeon®	Gold 5217	
Intel®	Xeon®	Gold 5218	
Intel®	Xeon®	Gold 5218B	
Intel®	Xeon®	Gold 5218N	
Intel®	Xeon®	Gold 5218R	
Intel®	Xeon®	Gold 5218T	
Intel®	Xeon®	Gold 5220	
Intel®	Xeon®	Gold 5220R	
Intel®	Xeon®	Gold 5220S	
Intel®	Xeon®	Gold 5220T	
Intel®	Xeon®	Gold 5222	
Intel®	Xeon®	Gold 5315Y	
Intel®	Xeon®	Gold 5317	
Intel®	Xeon®	Gold 5318N	
Intel®	Xeon®	Gold 5318S	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 5320	
Intel®	Xeon®	Gold 5320T	
Intel®	Xeon®	Gold 6126	
Intel®	Xeon®	Gold 6126F	
Intel®	Xeon®	Gold 6126T	
Intel®	Xeon®	Gold 6128	
Intel®	Xeon®	Gold 6130	
Intel®	Xeon®	Gold 6130F	
Intel®	Xeon®	Gold 6130T	
Intel®	Xeon®	Gold 6132	
Intel®	Xeon®	Gold 6134	
Intel®	Xeon®	Gold 6136	
Intel®	Xeon®	Gold 6138	
Intel®	Xeon®	Gold 6138F	
Intel®	Xeon®	Gold 6138P	
Intel®	Xeon®	Gold 6138T	
Intel®	Xeon®	Gold 6140	
Intel®	Xeon®	Gold 6142	
Intel®	Xeon®	Gold 6142F	
Intel®	Xeon®	Gold 6144	
Intel®	Xeon®	Gold 6146	
Intel®	Xeon®	Gold 6148	
Intel®	Xeon®	Gold 6148F	
Intel®	Xeon®	Gold 6150	
Intel®	Xeon®	Gold 6152	
Intel®	Xeon®	Gold 6154	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6208U	
Intel®	Xeon®	Gold 6209U	
Intel®	Xeon®	Gold 6210U	
Intel®	Xeon®	Gold 6212U	
Intel®	Xeon®	Gold 6222V	
Intel®	Xeon®	Gold 6226	
Intel®	Xeon®	Gold 6226R	
Intel®	Xeon®	Gold 6230	
Intel®	Xeon®	Gold 6230N	
Intel®	Xeon®	Gold 6230R	
Intel®	Xeon®	Gold 6230T	
Intel®	Xeon®	Gold 6234	
Intel®	Xeon®	Gold 6238	
Intel®	Xeon®	Gold 6238L	
Intel®	Xeon®	Gold 6238R	
Intel®	Xeon®	Gold 6238T	
Intel®	Xeon®	Gold 6240	
Intel®	Xeon®	Gold 6240L	
Intel®	Xeon®	Gold 6240R	
Intel®	Xeon®	Gold 6240Y	
Intel®	Xeon®	Gold 6242	
Intel®	Xeon®	Gold 6242R	
Intel®	Xeon®	Gold 6244	
Intel®	Xeon®	Gold 6246	
Intel®	Xeon®	Gold 6246R	
Intel®	Xeon®	Gold 6248	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6248R	
Intel®	Xeon®	Gold 6250	
Intel®	Xeon®	Gold 6250L	
Intel®	Xeon®	Gold 6252	
Intel®	Xeon®	Gold 6252N	
Intel®	Xeon®	Gold 6254	
Intel®	Xeon®	Gold 6256	
Intel®	Xeon®	Gold 6258R	
Intel®	Xeon®	Gold 6262V	
Intel®	Xeon®	Gold 6312U	
Intel®	Xeon®	Gold 6314U	
Intel®	Xeon®	Gold 6326	
Intel®	Xeon®	Gold 6330	
Intel®	Xeon®	Gold 6330N	
Intel®	Xeon®	Gold 6334	
Intel®	Xeon®	Gold 6336Y	
Intel®	Xeon®	Gold 6338	
Intel®	Xeon®	Gold 6338N	
Intel®	Xeon®	Gold 6338T	
Intel®	Xeon®	Gold 6342	
Intel®	Xeon®	Gold 6346	
Intel®	Xeon®	Gold 6348	
Intel®	Xeon®	Gold 6354	
Intel®	Xeon®	Gold Gold 5318Y	
Intel®	Xeon®	Platinum 8153	
Intel®	Xeon®	Platinum 8156	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Platinum 8158	
Intel®	Xeon®	Platinum 8160	
Intel®	Xeon®	Platinum 8160F	
Intel®	Xeon®	Platinum 8160T	
Intel®	Xeon®	Platinum 8164	
Intel®	Xeon®	Platinum 8168	
Intel®	Xeon®	Platinum 8170	
Intel®	Xeon®	Platinum 8176	
Intel®	Xeon®	Platinum 8176F	
Intel®	Xeon®	Platinum 8180	
Intel®	Xeon®	Platinum 8253	
Intel®	Xeon®	Platinum 8256	
Intel®	Xeon®	Platinum 8260	
Intel®	Xeon®	Platinum 8260L	
Intel®	Xeon®	Platinum 8260Y	
Intel®	Xeon®	Platinum 8268	
Intel®	Xeon®	Platinum 8270	
Intel®	Xeon®	Platinum 8276	
Intel®	Xeon®	Platinum 8276L	
Intel®	Xeon®	Platinum 8280	
Intel®	Xeon®	Platinum 8280L	
Intel®	Xeon®	Platinum 8351N	
Intel®	Xeon®	Platinum 8352M	
Intel®	Xeon®	Platinum 8352S	
Intel®	Xeon®	Platinum 8352V	
Intel®	Xeon®	Platinum 8352Y	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Platinum 8358	
Intel®	Xeon®	Platinum 8358P	
Intel®	Xeon®	Platinum 8360Y	
Intel®	Xeon®	Platinum 8362	
Intel®	Xeon®	Platinum 8368	
Intel®	Xeon®	Platinum 8368Q	
Intel®	Xeon®	Platinum 8380	
Intel®	Xeon®	Platinum 9221	
Intel®	Xeon®	Platinum 9222	
Intel®	Xeon®	Platinum 9242	
Intel®	Xeon®	Platinum 9282	
Intel®	Xeon®	Silver 4108	
Intel®	Xeon®	Silver 4109T	
Intel®	Xeon®	Silver 4110	
Intel®	Xeon®	Silver 4112	
Intel®	Xeon®	Silver 4114	
Intel®	Xeon®	Silver 4114T	
Intel®	Xeon®	Silver 4116	
Intel®	Xeon®	Silver 4116T	
Intel®	Xeon®	Silver 4208	
Intel®	Xeon®	Silver 4209T	
Intel®	Xeon®	Silver 4210	
Intel®	Xeon®	Silver 4210R	
Intel®	Xeon®	Silver 4210T	
Intel®	Xeon®	Silver 4214	
Intel®	Xeon®	Silver 4214R	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Silver 4214Y	
Intel®	Xeon®	Silver 4215	
Intel®	Xeon®	Silver 4215R	
Intel®	Xeon®	Silver 4216	
Intel®	Xeon®	Silver 4309Y	
Intel®	Xeon®	Silver 4310	
Intel®	Xeon®	Silver 4310T	
Intel®	Xeon®	Silver 4314	
Intel®	Xeon®	Silver 4316	
Intel®	Xeon®	W-10855M	
Intel®	Xeon®	W-10885M	
Intel®	Xeon®	W-11155MLE	
Intel®	Xeon®	W-11155MRE	
Intel®	Xeon®	W-11555MLE	
Intel®	Xeon®	W-11555MRE	
Intel®	Xeon®	W-11855M	
Intel®	Xeon®	W-11865MLE	
Intel®	Xeon®	W-11865MRE	
Intel®	Xeon®	W-11955M	
Intel®	Xeon®	W-1250	
Intel®	Xeon®	W-1250E	
Intel®	Xeon®	W-1250P	
Intel®	Xeon®	W-1250TE	
Intel®	Xeon®	W-1270	
Intel®	Xeon®	W-1270E	
Intel®	Xeon®	W-1270P	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-1270TE	
Intel®	Xeon®	W-1290	
Intel®	Xeon®	W-1290E	
Intel®	Xeon®	W-1290P	
Intel®	Xeon®	W-1290T	
Intel®	Xeon®	W-1290TE	
Intel®	Xeon®	W-1350	
Intel®	Xeon®	W-1350P	
Intel®	Xeon®	W-1370	
Intel®	Xeon®	W-1370P	
Intel®	Xeon®	W-1390	
Intel®	Xeon®	W-1390P	
Intel®	Xeon®	W-1390T	
Intel®	Xeon®	W-2102	
Intel®	Xeon®	W-2104	
Intel®	Xeon®	W-2123	
Intel®	Xeon®	W-2125	
Intel®	Xeon®	W-2133	
Intel®	Xeon®	W-2135	
Intel®	Xeon®	W-2145	
Intel®	Xeon®	W-2155	
Intel®	Xeon®	W-2175	
Intel®	Xeon®	W-2195	
Intel®	Xeon®	W-2223	
Intel®	Xeon®	W-2225	
Intel®	Xeon®	W-2235	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-2245	
Intel®	Xeon®	W-2255	
Intel®	Xeon®	W-2265	
Intel®	Xeon®	W-2275	
Intel®	Xeon®	W-2295	
Intel®	Xeon®	W-3175X	
Intel®	Xeon®	W-3223	
Intel®	Xeon®	W-3225	
Intel®	Xeon®	W-3235	
Intel®	Xeon®	W-3245	
Intel®	Xeon®	W-3245M	
Intel®	Xeon®	W-3265	
Intel®	Xeon®	W-3265M	
Intel®	Xeon®	W-3275	
Intel®	Xeon®	W-3275M	

Windows 10 22H2 Supported Qualcomm Processors

Article • 05/23/2023

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Qualcomm®	Snapdragon™	Snapdragon 8cx
Qualcomm®	Snapdragon™	Snapdragon 8cx Gen 2

Windows 10 LTSB 1507 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D

Manufacturer	Brand	Model
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
	Series	
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU

Windows 10 LTSB 1507 Supported Intel Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
Intel®	Celeron®	3205U
Intel®	Celeron®	3215U
Intel®	Celeron®	3755U
Intel®	Celeron®	3765U
Intel®	Celeron®	3855U
Intel®	Celeron®	3955U
Intel®	Celeron®	G3900
Intel®	Celeron®	G3900E
Intel®	Celeron®	G3900T
Intel®	Celeron®	G3900TE
Intel®	Celeron®	G3902E
Intel®	Celeron®	G3920
Intel®	Core™	i3-5005U
Intel®	Core™	i3-5010U
Intel®	Core™	i3-5015U
Intel®	Core™	i3-5020U
Intel®	Core™	i3-5157U
Intel®	Core™	i3-6006U
Intel®	Core™	i3-6098P

Manufacturer	Brand	Model
Intel®	Core™	i3-6100
Intel®	Core™	i3-6100E
Intel®	Core™	i3-6100H
Intel®	Core™	i3-6100T
Intel®	Core™	i3-6100TE
Intel®	Core™	i3-6100U
Intel®	Core™	i3-6102E
Intel®	Core™	i3-6157U
Intel®	Core™	i3-6167U
Intel®	Core™	i3-6300
Intel®	Core™	i3-6300T
Intel®	Core™	i3-6320
Intel®	Core™	i5-5200U
Intel®	Core™	i5-5250U
Intel®	Core™	i5-5257U
Intel®	Core™	i5-5287U
Intel®	Core™	i5-5300U
Intel®	Core™	i5-5350H
Intel®	Core™	i5-5350U
Intel®	Core™	i5-5575R
Intel®	Core™	i5-5675C
Intel®	Core™	i5-5675R
Intel®	Core™	i5-6200U
Intel®	Core™	i5-6260U
Intel®	Core™	i5-6267U
Intel®	Core™	i5-6287U

Manufacturer	Brand	Model
Intel®	Core™	i5-6300HQ
Intel®	Core™	i5-6300U
Intel®	Core™	i5-6350HQ
Intel®	Core™	i5-6360U
Intel®	Core™	i5-6400
Intel®	Core™	i5-6400T
Intel®	Core™	i5-6402P
Intel®	Core™	i5-6440EQ
Intel®	Core™	i5-6440HQ
Intel®	Core™	i5-6442EQ
Intel®	Core™	i5-6500
Intel®	Core™	i5-6500T
Intel®	Core™	i5-6500TE
Intel®	Core™	i5-6585R
Intel®	Core™	i5-6600
Intel®	Core™	i5-6600K
Intel®	Core™	i5-6600T
Intel®	Core™	i5-6685R
Intel®	Core™	i7-5500U
Intel®	Core™	i7-5550U
Intel®	Core™	i7-5557U
Intel®	Core™	i7-5600U
Intel®	Core™	i7-5650U
Intel®	Core™	i7-5700EQ
Intel®	Core™	i7-5700HQ
Intel®	Core™	i7-5750HQ

Manufacturer	Brand	Model
Intel®	Core™	i7-5775C
Intel®	Core™	i7-5775R
Intel®	Core™	i7-5850EQ
Intel®	Core™	i7-5850HQ
Intel®	Core™	i7-5950HQ
Intel®	Core™	i7-6500U
Intel®	Core™	i7-6560U
Intel®	Core™	i7-6567U
Intel®	Core™	i7-6600U
Intel®	Core™	i7-6650U
Intel®	Core™	i7-6660U
Intel®	Core™	i7-6700
Intel®	Core™	i7-6700HQ
Intel®	Core™	i7-6700K
Intel®	Core™	i7-6700T
Intel®	Core™	i7-6700TE
Intel®	Core™	i7-6770HQ
Intel®	Core™	i7-6785R
Intel®	Core™	i7-6820EQ
Intel®	Core™	i7-6820HK
Intel®	Core™	i7-6820HQ
Intel®	Core™	i7-6822EQ
Intel®	Core™	i7-6870HQ
Intel®	Core™	i7-6920HQ
Intel®	Core™	i7-6970HQ
Intel®	Core™	i7-7800X

Manufacturer	Brand	Model
Intel®	Core™	i7-7820X
Intel®	Core™	i7-9800X
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-6Y30
Intel®	Core™	m5-6Y54
Intel®	Core™	m5-6Y57
Intel®	Core™	M-5Y10
Intel®	Core™	M-5Y10a
Intel®	Core™	M-5Y10c
Intel®	Core™	M-5Y31
Intel®	Core™	M-5Y51
Intel®	Core™	M-5Y70
Intel®	Core™	M-5Y71
Intel®	Core™	m7-6Y75
Intel®	Pentium®	3805U
Intel®	Pentium®	3825U

Manufacturer	Brand	Model
Intel®	Pentium®	4405U
Intel®	Pentium®	4405Y
Intel®	Pentium®	D1507
Intel®	Pentium®	D1508
Intel®	Pentium®	D1509
Intel®	Pentium®	D1517
Intel®	Pentium®	D1519
Intel®	Pentium®	G4400
Intel®	Pentium®	G4400T
Intel®	Pentium®	G4400TE
Intel®	Pentium®	G4500
Intel®	Pentium®	G4500T
Intel®	Pentium®	G4520
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	D-1513N
Intel®	Xeon®	D-1518
Intel®	Xeon®	D-1520
Intel®	Xeon®	D-1521
Intel®	Xeon®	D-1523N
Intel®	Xeon®	D-1527
Intel®	Xeon®	D-1528
Intel®	Xeon®	D-1529
Intel®	Xeon®	D-1531
Intel®	Xeon®	D-1533N
Intel®	Xeon®	D-1537

Manufacturer	Brand	Model
Intel®	Xeon®	D-1539
Intel®	Xeon®	D-1540
Intel®	Xeon®	D-1541
Intel®	Xeon®	D-1543N
Intel®	Xeon®	D-1548
Intel®	Xeon®	D-1553N
Intel®	Xeon®	D-1557
Intel®	Xeon®	D-1559
Intel®	Xeon®	D-1567
Intel®	Xeon®	D-1571
Intel®	Xeon®	D-1577
Intel®	Xeon®	D-2123IT
Intel®	Xeon®	D-2141I
Intel®	Xeon®	D-2142IT
Intel®	Xeon®	D-2143IT
Intel®	Xeon®	D-2145NT
Intel®	Xeon®	D-2146NT
Intel®	Xeon®	D-2161I
Intel®	Xeon®	D-2163IT
Intel®	Xeon®	D-2166NT
Intel®	Xeon®	D-2173IT
Intel®	Xeon®	D-2177NT
Intel®	Xeon®	D-2183IT
Intel®	Xeon®	D-2187NT
Intel®	Xeon®	E3-1220 v5
Intel®	Xeon®	E3-1225 v5

Manufacturer	Brand	Model
Intel®	Xeon®	E3-1230 v5
Intel®	Xeon®	E3-1235L v5
Intel®	Xeon®	E3-1240 v5
Intel®	Xeon®	E3-1240L v5
Intel®	Xeon®	E3-1245 v5
Intel®	Xeon®	E3-1258L v4
Intel®	Xeon®	E3-1260L v5
Intel®	Xeon®	E3-1265L v4
Intel®	Xeon®	E3-1268L v5
Intel®	Xeon®	E3-1270 v5
Intel®	Xeon®	E3-1275 v5
Intel®	Xeon®	E3-1278L v4
Intel®	Xeon®	E3-1280 v5
Intel®	Xeon®	E3-1285 v4
Intel®	Xeon®	E3-1285L v4
Intel®	Xeon®	E3-1505L v5
Intel®	Xeon®	E3-1505M v5
Intel®	Xeon®	E3-1515M v5
Intel®	Xeon®	E3-1535M v5
Intel®	Xeon®	E3-1545M v5
Intel®	Xeon®	E3-1558L v5
Intel®	Xeon®	E3-1565L v5
Intel®	Xeon®	E3-1575M v5
Intel®	Xeon®	E3-1578L v5
Intel®	Xeon®	E3-1585 v5
Intel®	Xeon®	E3-1585L v5

Manufacturer	Brand	Model
Intel®	Xeon®	E5-1620 v4
Intel®	Xeon®	E5-1630 v4
Intel®	Xeon®	E5-1650 v4
Intel®	Xeon®	E5-1660 v4
Intel®	Xeon®	E5-1680 v4
Intel®	Xeon®	E5-2603 v4
Intel®	Xeon®	E5-2608L v4
Intel®	Xeon®	E5-2609 v4
Intel®	Xeon®	E5-2618L v4
Intel®	Xeon®	E5-2620 v4
Intel®	Xeon®	E5-2623 v4
Intel®	Xeon®	E5-2628L v4
Intel®	Xeon®	E5-2630 v4
Intel®	Xeon®	E5-2630L v4
Intel®	Xeon®	E5-2637 v4
Intel®	Xeon®	E5-2640 v4
Intel®	Xeon®	E5-2643 v4
Intel®	Xeon®	E5-2648L v4
Intel®	Xeon®	E5-2650 v4
Intel®	Xeon®	E5-2650L v4
Intel®	Xeon®	E5-2658 v4
Intel®	Xeon®	E5-2660 v4
Intel®	Xeon®	E5-2667 v4
Intel®	Xeon®	E5-2680 v4
Intel®	Xeon®	E5-2683 v4
Intel®	Xeon®	E5-2687W v4

Manufacturer	Brand	Model
Intel®	Xeon®	E5-2690 v4
Intel®	Xeon®	E5-2695 v4
Intel®	Xeon®	E5-2697 v4
Intel®	Xeon®	E5-2697A v4
Intel®	Xeon®	E5-2698 v4
Intel®	Xeon®	E5-2699 v4
Intel®	Xeon®	E5-2699A v4
Intel®	Xeon®	E5-2699R v4
Intel®	Xeon®	E5-4610 v4
Intel®	Xeon®	E5-4620 v4
Intel®	Xeon®	E5-4627 v4
Intel®	Xeon®	E5-4628L v4
Intel®	Xeon®	E5-4640 v4
Intel®	Xeon®	E5-4650 v4
Intel®	Xeon®	E5-4655 v4
Intel®	Xeon®	E5-4660 v4
Intel®	Xeon®	E5-4667 v4
Intel®	Xeon®	E5-4669 v4
Intel®	Xeon®	E7-4809 v4
Intel®	Xeon®	E7-4820 v4
Intel®	Xeon®	E7-4830 v4
Intel®	Xeon®	E7-4850 v4
Intel®	Xeon®	E7-8860 v4
Intel®	Xeon®	E7-8867 v4
Intel®	Xeon®	E7-8870 v4
Intel®	Xeon®	E7-8880 v4

Manufacturer	Brand	Model
Intel®	Xeon®	E7-8890 v4
Intel®	Xeon®	E7-8891 v4
Intel®	Xeon®	E7-8893 v4
Intel®	Xeon®	E7-8894 v4
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-3175X

Windows 10 LTSB 1607 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler

Manufacturer	Brand	Model
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D

Manufacturer	Brand	Model
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320

Manufacturer	Brand	Model
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-	A10 PRO-7850B with Radeon™ R7 Graphics

Manufacturer	Brand	Model
	Series	
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B

Windows 10 LTSB 1607 Supported Intel Processors

Article • 07/15/2022

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Intel®	Atom®	E3805	IoT Enterprise Only
Intel®	Atom®	E3827	IoT Enterprise Only
Intel®	Atom®	E3826	IoT Enterprise Only
Intel®	Atom®	E3815	IoT Enterprise Only
Intel®	Atom®	E3845	IoT Enterprise Only
Intel®	Atom®	E3825	IoT Enterprise Only
Intel®	Atom®	x5-E3930	IoT Enterprise Only
Intel®	Atom®	x5-E3940	IoT Enterprise Only
Intel®	Atom®	x7-E3950	IoT Enterprise Only
Intel®	Atom®	x5-E8000	IoT Enterprise Only
Intel®	Celeron®	2000E	IoT Enterprise Only
Intel®	Celeron®	2002E	IoT Enterprise Only
Intel®	Celeron®	2950M	IoT Enterprise Only
Intel®	Celeron®	2955U	IoT Enterprise Only
Intel®	Celeron®	2957U	IoT Enterprise Only
Intel®	Celeron®	2961Y	IoT Enterprise Only
Intel®	Celeron®	2970M	IoT Enterprise Only
Intel®	Celeron®	2980U	IoT Enterprise Only
Intel®	Celeron®	2981U	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	3205U	
Intel®	Celeron®	3215U	
Intel®	Celeron®	3755U	
Intel®	Celeron®	3765U	
Intel®	Celeron®	3855U	
Intel®	Celeron®	3865U	
Intel®	Celeron®	3955U	
Intel®	Celeron®	3965U	
Intel®	Celeron®	3965Y	
Intel®	Celeron®	G1820	IoT Enterprise Only
Intel®	Celeron®	G1820T	IoT Enterprise Only
Intel®	Celeron®	G1820TE	IoT Enterprise Only
Intel®	Celeron®	G1830	IoT Enterprise Only
Intel®	Celeron®	G1840	IoT Enterprise Only
Intel®	Celeron®	G1840T	IoT Enterprise Only
Intel®	Celeron®	G1850	IoT Enterprise Only
Intel®	Celeron®	G3900	
Intel®	Celeron®	G3900E	
Intel®	Celeron®	G3900T	
Intel®	Celeron®	G3900TE	
Intel®	Celeron®	G3902E	
Intel®	Celeron®	G3920	
Intel®	Celeron®	G3930	
Intel®	Celeron®	G3930E	
Intel®	Celeron®	G3930T	
Intel®	Celeron®	G3930TE	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	G3950	
Intel®	Celeron®	J1900	IoT Enterprise Only
Intel®	Celeron®	J3355	IoT Enterprise Only
Intel®	Celeron®	J3355E	IoT Enterprise Only
Intel®	Celeron®	J3455	IoT Enterprise Only
Intel®	Celeron®	J3455E	IoT Enterprise Only
Intel®	Celeron®	N2807	IoT Enterprise Only
Intel®	Celeron®	N2920	IoT Enterprise Only
Intel®	Celeron®	N2930	IoT Enterprise Only
Intel®	Celeron®	N3010	IoT Enterprise Only
Intel®	Celeron®	N3060	IoT Enterprise Only
Intel®	Celeron®	N3160	IoT Enterprise Only
Intel®	Celeron®	N3350	IoT Enterprise Only
Intel®	Core™	M-5Y10	
Intel®	Core™	M-5Y10a	
Intel®	Core™	M-5Y10c	
Intel®	Core™	M-5Y31	
Intel®	Core™	M-5Y51	
Intel®	Core™	M-5Y70	
Intel®	Core™	M-5Y71	
Intel®	Core™	i3-4000M	IoT Enterprise Only
Intel®	Core™	i3-4005U	IoT Enterprise Only
Intel®	Core™	i3-4010U	IoT Enterprise Only
Intel®	Core™	i3-4010Y	IoT Enterprise Only
Intel®	Core™	i3-4012Y	IoT Enterprise Only
Intel®	Core™	i3-4020Y	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-4025U	IoT Enterprise Only
Intel®	Core™	i3-4030U	IoT Enterprise Only
Intel®	Core™	i3-4030Y	IoT Enterprise Only
Intel®	Core™	i3-4100E	IoT Enterprise Only
Intel®	Core™	i3-4100M	IoT Enterprise Only
Intel®	Core™	i3-4100U	IoT Enterprise Only
Intel®	Core™	i3-4102E	IoT Enterprise Only
Intel®	Core™	i3-4110E	IoT Enterprise Only
Intel®	Core™	i3-4110M	IoT Enterprise Only
Intel®	Core™	i3-4112E	IoT Enterprise Only
Intel®	Core™	i3-4120U	IoT Enterprise Only
Intel®	Core™	i3-4130	IoT Enterprise Only
Intel®	Core™	i3-4130T	IoT Enterprise Only
Intel®	Core™	i3-4150	IoT Enterprise Only
Intel®	Core™	i3-4150T	IoT Enterprise Only
Intel®	Core™	i3-4158U	IoT Enterprise Only
Intel®	Core™	i3-4160	IoT Enterprise Only
Intel®	Core™	i3-4160T	IoT Enterprise Only
Intel®	Core™	i3-4170	IoT Enterprise Only
Intel®	Core™	i3-4170T	IoT Enterprise Only
Intel®	Core™	i3-4330	IoT Enterprise Only
Intel®	Core™	i3-4330T	IoT Enterprise Only
Intel®	Core™	i3-4330TE	IoT Enterprise Only
Intel®	Core™	i3-4340	IoT Enterprise Only
Intel®	Core™	i3-4340TE	IoT Enterprise Only
Intel®	Core™	i3-4350	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-4350T	IoT Enterprise Only
Intel®	Core™	i3-4360	IoT Enterprise Only
Intel®	Core™	i3-4360T	IoT Enterprise Only
Intel®	Core™	i3-4370	IoT Enterprise Only
Intel®	Core™	i3-4370T	IoT Enterprise Only
Intel®	Core™	i3-5005U	
Intel®	Core™	i3-5010U	
Intel®	Core™	i3-5015U	
Intel®	Core™	i3-5020U	
Intel®	Core™	i3-5157U	
Intel®	Core™	i3-6006U	
Intel®	Core™	i3-6098P	
Intel®	Core™	i3-6100	
Intel®	Core™	i3-6100E	
Intel®	Core™	i3-6100H	
Intel®	Core™	i3-6100T	
Intel®	Core™	i3-6100TE	
Intel®	Core™	i3-6100U	
Intel®	Core™	i3-6102E	
Intel®	Core™	i3-6157U	
Intel®	Core™	i3-6167U	
Intel®	Core™	i3-6300	
Intel®	Core™	i3-6300T	
Intel®	Core™	i3-6320	
Intel®	Core™	i3-7020U	
Intel®	Core™	i3-7100	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-7100E	
Intel®	Core™	i3-7100H	
Intel®	Core™	i3-7100T	
Intel®	Core™	i3-7100U	
Intel®	Core™	i3-7101E	
Intel®	Core™	i3-7101TE	
Intel®	Core™	i3-7102E	
Intel®	Core™	i3-7130U	
Intel®	Core™	i3-7167U	
Intel®	Core™	i3-7300	
Intel®	Core™	i3-7300T	
Intel®	Core™	i3-7320	
Intel®	Core™	i3-7350K	
Intel®	Core™	i5-4200H	IoT Enterprise Only
Intel®	Core™	i5-4200M	IoT Enterprise Only
Intel®	Core™	i5-4200U	IoT Enterprise Only
Intel®	Core™	i5-4200Y	IoT Enterprise Only
Intel®	Core™	i5-4202Y	IoT Enterprise Only
Intel®	Core™	i5-4210H	IoT Enterprise Only
Intel®	Core™	i5-4210M	IoT Enterprise Only
Intel®	Core™	i5-4210U	IoT Enterprise Only
Intel®	Core™	i5-4210Y	IoT Enterprise Only
Intel®	Core™	i5-4220Y	IoT Enterprise Only
Intel®	Core™	i5-4250U	IoT Enterprise Only
Intel®	Core™	i5-4258U	IoT Enterprise Only
Intel®	Core™	i5-4260U	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-4278U	IoT Enterprise Only
Intel®	Core™	i5-4288U	IoT Enterprise Only
Intel®	Core™	i5-4300M	IoT Enterprise Only
Intel®	Core™	i5-4300U	IoT Enterprise Only
Intel®	Core™	i5-4300Y	IoT Enterprise Only
Intel®	Core™	i5-4302Y	IoT Enterprise Only
Intel®	Core™	i5-4308U	IoT Enterprise Only
Intel®	Core™	i5-4310M	IoT Enterprise Only
Intel®	Core™	i5-4310U	IoT Enterprise Only
Intel®	Core™	i5-4330M	IoT Enterprise Only
Intel®	Core™	i5-4340M	IoT Enterprise Only
Intel®	Core™	i5-4350U	IoT Enterprise Only
Intel®	Core™	i5-4360U	IoT Enterprise Only
Intel®	Core™	i5-4400E	IoT Enterprise Only
Intel®	Core™	i5-4402E	IoT Enterprise Only
Intel®	Core™	i5-4402EC	IoT Enterprise Only
Intel®	Core™	i5-4410E	IoT Enterprise Only
Intel®	Core™	i5-4422E	IoT Enterprise Only
Intel®	Core™	i5-4430	IoT Enterprise Only
Intel®	Core™	i5-4430S	IoT Enterprise Only
Intel®	Core™	i5-4440	IoT Enterprise Only
Intel®	Core™	i5-4440S	IoT Enterprise Only
Intel®	Core™	i5-4460	IoT Enterprise Only
Intel®	Core™	i5-4460S	IoT Enterprise Only
Intel®	Core™	i5-4460T	IoT Enterprise Only
Intel®	Core™	i5-4570	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-4570S	IoT Enterprise Only
Intel®	Core™	i5-4570T	IoT Enterprise Only
Intel®	Core™	i5-4570TE	IoT Enterprise Only
Intel®	Core™	i5-4590	IoT Enterprise Only
Intel®	Core™	i5-4590S	IoT Enterprise Only
Intel®	Core™	i5-4590T	IoT Enterprise Only
Intel®	Core™	i5-4670	IoT Enterprise Only
Intel®	Core™	i5-4670K	IoT Enterprise Only
Intel®	Core™	i5-4670S	IoT Enterprise Only
Intel®	Core™	i5-4670T	IoT Enterprise Only
Intel®	Core™	i5-4690	IoT Enterprise Only
Intel®	Core™	i5-4690S	IoT Enterprise Only
Intel®	Core™	i5-4690T	IoT Enterprise Only
Intel®	Core™	i5-5200U	
Intel®	Core™	i5-5250U	
Intel®	Core™	i5-5257U	
Intel®	Core™	i5-5287U	
Intel®	Core™	i5-5300U	
Intel®	Core™	i5-5350H	
Intel®	Core™	i5-5350U	
Intel®	Core™	i5-5575R	
Intel®	Core™	i5-5675C	
Intel®	Core™	i5-5675R	
Intel®	Core™	i5-6200U	
Intel®	Core™	i5-6260U	
Intel®	Core™	i5-6267U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-6287U	
Intel®	Core™	i5-6300HQ	
Intel®	Core™	i5-6300U	
Intel®	Core™	i5-6350HQ	
Intel®	Core™	i5-6360U	
Intel®	Core™	i5-6400	
Intel®	Core™	i5-6400T	
Intel®	Core™	i5-6402P	
Intel®	Core™	i5-6440EQ	
Intel®	Core™	i5-6440HQ	
Intel®	Core™	i5-6442EQ	
Intel®	Core™	i5-6500	
Intel®	Core™	i5-6500T	
Intel®	Core™	i5-6500TE	
Intel®	Core™	i5-6585R	
Intel®	Core™	i5-6600	
Intel®	Core™	i5-6600K	
Intel®	Core™	i5-6600T	
Intel®	Core™	i5-6685R	
Intel®	Core™	i5-7200U	
Intel®	Core™	i5-7260U	
Intel®	Core™	i5-7267U	
Intel®	Core™	i5-7287U	
Intel®	Core™	i5-7300HQ	
Intel®	Core™	i5-7300U	
Intel®	Core™	i5-7360U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-7400	
Intel®	Core™	i5-7400T	
Intel®	Core™	i5-7440EQ	
Intel®	Core™	i5-7440HQ	
Intel®	Core™	i5-7442EQ	
Intel®	Core™	i5-7500	
Intel®	Core™	i5-7500T	
Intel®	Core™	i5-7600	
Intel®	Core™	i5-7600K	
Intel®	Core™	i5-7600T	
Intel®	Core™	i5-7640X	
Intel®	Core™	i5-7Y54	
Intel®	Core™	i5-7Y57	
Intel®	Core™	i7-4500U	IoT Enterprise Only
Intel®	Core™	i7-4510U	IoT Enterprise Only
Intel®	Core™	i7-4550U	IoT Enterprise Only
Intel®	Core™	i7-4558U	IoT Enterprise Only
Intel®	Core™	i7-4578U	IoT Enterprise Only
Intel®	Core™	i7-4600M	IoT Enterprise Only
Intel®	Core™	i7-4600U	IoT Enterprise Only
Intel®	Core™	i7-4610M	IoT Enterprise Only
Intel®	Core™	i7-4610Y	IoT Enterprise Only
Intel®	Core™	i7-4650U	IoT Enterprise Only
Intel®	Core™	i7-4700EC	IoT Enterprise Only
Intel®	Core™	i7-4700EQ	IoT Enterprise Only
Intel®	Core™	i7-4700HQ	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-4700MQ	IoT Enterprise Only
Intel®	Core™	i7-4702EC	IoT Enterprise Only
Intel®	Core™	i7-4702HQ	IoT Enterprise Only
Intel®	Core™	i7-4702MQ	IoT Enterprise Only
Intel®	Core™	i7-4710HQ	IoT Enterprise Only
Intel®	Core™	i7-4710MQ	IoT Enterprise Only
Intel®	Core™	i7-4712HQ	IoT Enterprise Only
Intel®	Core™	i7-4712MQ	IoT Enterprise Only
Intel®	Core™	i7-4720HQ	IoT Enterprise Only
Intel®	Core™	i7-4722HQ	IoT Enterprise Only
Intel®	Core™	i7-4765T	IoT Enterprise Only
Intel®	Core™	i7-4770	IoT Enterprise Only
Intel®	Core™	i7-4770K	IoT Enterprise Only
Intel®	Core™	i7-4770S	IoT Enterprise Only
Intel®	Core™	i7-4770T	IoT Enterprise Only
Intel®	Core™	i7-4770TE	IoT Enterprise Only
Intel®	Core™	i7-4771	IoT Enterprise Only
Intel®	Core™	i7-4785T	IoT Enterprise Only
Intel®	Core™	i7-4790	IoT Enterprise Only
Intel®	Core™	i7-4790S	IoT Enterprise Only
Intel®	Core™	i7-4790T	IoT Enterprise Only
Intel®	Core™	i7-4800MQ	IoT Enterprise Only
Intel®	Core™	i7-4810MQ	IoT Enterprise Only
Intel®	Core™	i7-4900MQ	IoT Enterprise Only
Intel®	Core™	i7-4910MQ	IoT Enterprise Only
Intel®	Core™	i7-4930MX	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-4940MX	IoT Enterprise Only
Intel®	Core™	i7-5500U	
Intel®	Core™	i7-5550U	
Intel®	Core™	i7-5557U	
Intel®	Core™	i7-5600U	
Intel®	Core™	i7-5650U	
Intel®	Core™	i7-5700EQ	
Intel®	Core™	i7-5700HQ	
Intel®	Core™	i7-5750HQ	
Intel®	Core™	i7-5775C	
Intel®	Core™	i7-5775R	
Intel®	Core™	i7-5850EQ	
Intel®	Core™	i7-5850HQ	
Intel®	Core™	i7-5950HQ	
Intel®	Core™	i7-6500U	
Intel®	Core™	i7-6560U	
Intel®	Core™	i7-6567U	
Intel®	Core™	i7-6600U	
Intel®	Core™	i7-6650U	
Intel®	Core™	i7-6660U	
Intel®	Core™	i7-6700	
Intel®	Core™	i7-6700HQ	
Intel®	Core™	i7-6700K	
Intel®	Core™	i7-6700T	
Intel®	Core™	i7-6700TE	
Intel®	Core™	i7-6770HQ	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-6785R	
Intel®	Core™	i7-6820EQ	
Intel®	Core™	i7-6820HK	
Intel®	Core™	i7-6820HQ	
Intel®	Core™	i7-6822EQ	
Intel®	Core™	i7-6870HQ	
Intel®	Core™	i7-6920HQ	
Intel®	Core™	i7-6970HQ	
Intel®	Core™	i7-7500U	
Intel®	Core™	i7-7560U	
Intel®	Core™	i7-7567U	
Intel®	Core™	i7-7600U	
Intel®	Core™	i7-7660U	
Intel®	Core™	i7-7700	
Intel®	Core™	i7-7700HQ	
Intel®	Core™	i7-7700K	
Intel®	Core™	i7-7700T	
Intel®	Core™	i7-7740X	
Intel®	Core™	i7-7800X	
Intel®	Core™	i7-7820EQ	
Intel®	Core™	i7-7820HK	
Intel®	Core™	i7-7820HQ	
Intel®	Core™	i7-7820X	
Intel®	Core™	i7-7920HQ	
Intel®	Core™	i7-7Y75	
Intel®	Core™	i7-9800X	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-7900X	
Intel®	Core™	i9-7920X	
Intel®	Core™	i9-7940X	
Intel®	Core™	i9-7960X	
Intel®	Core™	i9-7980XE	
Intel®	Core™	i9-9820X	
Intel®	Core™	i9-9900X	
Intel®	Core™	i9-9920X	
Intel®	Core™	i9-9940X	
Intel®	Core™	i9-9960X	
Intel®	Core™	i9-9980XE	
Intel®	Core™	m3-6Y30	
Intel®	Core™	m3-7Y30	
Intel®	Core™	m3-7Y32	
Intel®	Core™	m5-6Y54	
Intel®	Core™	m5-6Y57	
Intel®	Core™	m7-6Y75	
Intel®	Pentium®	3550M	IoT Enterprise Only
Intel®	Pentium®	3556U	IoT Enterprise Only
Intel®	Pentium®	3558U	IoT Enterprise Only
Intel®	Pentium®	3560M	IoT Enterprise Only
Intel®	Pentium®	3560Y	IoT Enterprise Only
Intel®	Pentium®	3561Y	IoT Enterprise Only
Intel®	Pentium®	3805U	
Intel®	Pentium®	3825U	
Intel®	Pentium®	4405U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	4405Y	
Intel®	Pentium®	D1507	
Intel®	Pentium®	D1508	
Intel®	Pentium®	D1509	
Intel®	Pentium®	D1517	
Intel®	Pentium®	D1519	
Intel®	Pentium®	G3220	IoT Enterprise Only
Intel®	Pentium®	G3220T	IoT Enterprise Only
Intel®	Pentium®	G3240	IoT Enterprise Only
Intel®	Pentium®	G3240T	IoT Enterprise Only
Intel®	Pentium®	G3250	IoT Enterprise Only
Intel®	Pentium®	G3250T	IoT Enterprise Only
Intel®	Pentium®	G3258	IoT Enterprise Only
Intel®	Pentium®	G3260	IoT Enterprise Only
Intel®	Pentium®	G3260T	IoT Enterprise Only
Intel®	Pentium®	G3320TE	IoT Enterprise Only
Intel®	Pentium®	G3420	IoT Enterprise Only
Intel®	Pentium®	G3420T	IoT Enterprise Only
Intel®	Pentium®	G3430	IoT Enterprise Only
Intel®	Pentium®	G3440	IoT Enterprise Only
Intel®	Pentium®	G3440T	IoT Enterprise Only
Intel®	Pentium®	G3450	IoT Enterprise Only
Intel®	Pentium®	G3450T	IoT Enterprise Only
Intel®	Pentium®	G3460	IoT Enterprise Only
Intel®	Pentium®	G3460T	IoT Enterprise Only
Intel®	Pentium®	G3470	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	G4400	
Intel®	Pentium®	G4400T	
Intel®	Pentium®	G4400TE	
Intel®	Pentium®	G4500	
Intel®	Pentium®	G4500T	
Intel®	Pentium®	G4520	
Intel®	Pentium®	G4560	
Intel®	Pentium®	G4560T	
Intel®	Pentium®	G4600	
Intel®	Pentium®	G4600T	
Intel®	Pentium®	G4620	
Intel®	Pentium®	Gold 4410Y	
Intel®	Pentium®	Gold 4415U	
Intel®	Pentium®	Gold 4415Y	
Intel®	Pentium®	N3710	IoT Enterprise Only
Intel®	Pentium®	N4200	IoT Enterprise Only
Intel®	Xeon®	Bronze 3104	
Intel®	Xeon®	Bronze 3106	
Intel®	Xeon®	D-1513N	
Intel®	Xeon®	D-1518	
Intel®	Xeon®	D-1520	
Intel®	Xeon®	D-1521	
Intel®	Xeon®	D-1523N	
Intel®	Xeon®	D-1527	
Intel®	Xeon®	D-1528	
Intel®	Xeon®	D-1529	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	D-1531	
Intel®	Xeon®	D-1533N	
Intel®	Xeon®	D-1537	
Intel®	Xeon®	D-1539	
Intel®	Xeon®	D-1540	
Intel®	Xeon®	D-1541	
Intel®	Xeon®	D-1543N	
Intel®	Xeon®	D-1548	
Intel®	Xeon®	D-1553N	
Intel®	Xeon®	D-1557	
Intel®	Xeon®	D-1559	
Intel®	Xeon®	D-1567	
Intel®	Xeon®	D-1571	
Intel®	Xeon®	D-1577	
Intel®	Xeon®	D-2123IT	
Intel®	Xeon®	D-2141I	
Intel®	Xeon®	D-2142IT	
Intel®	Xeon®	D-2143IT	
Intel®	Xeon®	D-2145NT	
Intel®	Xeon®	D-2146NT	
Intel®	Xeon®	D-2161I	
Intel®	Xeon®	D-2163IT	
Intel®	Xeon®	D-2166NT	
Intel®	Xeon®	D-2173IT	
Intel®	Xeon®	D-2177NT	
Intel®	Xeon®	D-2183IT	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	D-2187NT	
Intel®	Xeon®	E3-1220 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1220 v5	
Intel®	Xeon®	E3-1220 v6	
Intel®	Xeon®	E3-1220L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1225 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1225 v5	
Intel®	Xeon®	E3-1225 v6	
Intel®	Xeon®	E3-1226 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1230 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1230 v5	
Intel®	Xeon®	E3-1230 v6	
Intel®	Xeon®	E3-1230L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1231 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1235L v5	
Intel®	Xeon®	E3-1240 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1240 v5	
Intel®	Xeon®	E3-1240 v6	
Intel®	Xeon®	E3-1240L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1240L v5	
Intel®	Xeon®	E3-1241 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1245 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1245 v5	
Intel®	Xeon®	E3-1245 v6	
Intel®	Xeon®	E3-1246 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1258L v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1260L v5	
Intel®	Xeon®	E3-1265L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1265L v4	
Intel®	Xeon®	E3-1268L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1268L v5	
Intel®	Xeon®	E3-1270 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1270 v5	
Intel®	Xeon®	E3-1270 v6	
Intel®	Xeon®	E3-1271 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1275 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1275 v5	
Intel®	Xeon®	E3-1275 v6	
Intel®	Xeon®	E3-1275L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1276 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1278L v4	
Intel®	Xeon®	E3-1280 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1280 v5	
Intel®	Xeon®	E3-1280 v6	
Intel®	Xeon®	E3-1281 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1285 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1285 v4	
Intel®	Xeon®	E3-1285 v6	
Intel®	Xeon®	E3-1285L v3	IoT Enterprise Only
Intel®	Xeon®	E3-1285L v4	
Intel®	Xeon®	E3-1286 v3	IoT Enterprise Only
Intel®	Xeon®	E3-1286L v3	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1501L v6	
Intel®	Xeon®	E3-1501M v6	
Intel®	Xeon®	E3-1505L v5	
Intel®	Xeon®	E3-1505L v6	
Intel®	Xeon®	E3-1505M v5	
Intel®	Xeon®	E3-1505M v6	
Intel®	Xeon®	E3-1515M v5	
Intel®	Xeon®	E3-1535M v5	
Intel®	Xeon®	E3-1535M v6	
Intel®	Xeon®	E3-1545M v5	
Intel®	Xeon®	E3-1558L v5	
Intel®	Xeon®	E3-1565L v5	
Intel®	Xeon®	E3-1575M v5	
Intel®	Xeon®	E3-1578L v5	
Intel®	Xeon®	E3-1585 v5	
Intel®	Xeon®	E3-1585L v5	
Intel®	Xeon®	E5-1428L v3	IoT Enterprise Only
Intel®	Xeon®	E5-1620 v3	IoT Enterprise Only
Intel®	Xeon®	E5-1620 v4	
Intel®	Xeon®	E5-1630 v3	IoT Enterprise Only
Intel®	Xeon®	E5-1630 v4	
Intel®	Xeon®	E5-1650 v3	IoT Enterprise Only
Intel®	Xeon®	E5-1650 v4	
Intel®	Xeon®	E5-1660 v3	IoT Enterprise Only
Intel®	Xeon®	E5-1660 v4	
Intel®	Xeon®	E5-1680 v3	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-1680 v4	
Intel®	Xeon®	E5-2408L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2418L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2428L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2438L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2603 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2603 v4	
Intel®	Xeon®	E5-2608L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2608L v4	
Intel®	Xeon®	E5-2609 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2609 v4	
Intel®	Xeon®	E5-2618L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2618L v4	
Intel®	Xeon®	E5-2620 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2620 v4	
Intel®	Xeon®	E5-2623 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2623 v4	
Intel®	Xeon®	E5-2628L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2628L v4	
Intel®	Xeon®	E5-2630 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2630 v4	
Intel®	Xeon®	E5-2630L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2630L v4	
Intel®	Xeon®	E5-2637 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2637 v4	
Intel®	Xeon®	E5-2640 v3	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-2640 v4	
Intel®	Xeon®	E5-2643 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2643 v4	
Intel®	Xeon®	E5-2648L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2648L v4	
Intel®	Xeon®	E5-2650 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2650 v4	
Intel®	Xeon®	E5-2650L v3	IoT Enterprise Only
Intel®	Xeon®	E5-2650L v4	
Intel®	Xeon®	E5-2658 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2658 v4	
Intel®	Xeon®	E5-2658A v3	IoT Enterprise Only
Intel®	Xeon®	E5-2660 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2660 v4	
Intel®	Xeon®	E5-2667 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2667 v4	
Intel®	Xeon®	E5-2670 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2680 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2680 v4	
Intel®	Xeon®	E5-2683 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2683 v4	
Intel®	Xeon®	E5-2687W v3	IoT Enterprise Only
Intel®	Xeon®	E5-2687W v4	
Intel®	Xeon®	E5-2690 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2690 v4	
Intel®	Xeon®	E5-2695 v3	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-2695 v4	
Intel®	Xeon®	E5-2697 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2697 v4	
Intel®	Xeon®	E5-2697A v4	
Intel®	Xeon®	E5-2698 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2698 v4	
Intel®	Xeon®	E5-2699 v3	IoT Enterprise Only
Intel®	Xeon®	E5-2699 v4	
Intel®	Xeon®	E5-2699A v4	
Intel®	Xeon®	E5-2699R v4	
Intel®	Xeon®	E5-4610 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4610 v4	
Intel®	Xeon®	E5-4620 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4620 v4	
Intel®	Xeon®	E5-4627 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4627 v4	
Intel®	Xeon®	E5-4628L v4	
Intel®	Xeon®	E5-4640 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4640 v4	
Intel®	Xeon®	E5-4648 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4650 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4650 v4	
Intel®	Xeon®	E5-4655 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4655 v4	
Intel®	Xeon®	E5-4660 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4660 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-4667 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4667 v4	
Intel®	Xeon®	E5-4669 v3	IoT Enterprise Only
Intel®	Xeon®	E5-4669 v4	
Intel®	Xeon®	E7-4809 v3	IoT Enterprise Only
Intel®	Xeon®	E7-4809 v4	
Intel®	Xeon®	E7-4820 v3	IoT Enterprise Only
Intel®	Xeon®	E7-4820 v4	
Intel®	Xeon®	E7-4830 v3	IoT Enterprise Only
Intel®	Xeon®	E7-4830 v4	
Intel®	Xeon®	E7-4850 v3	IoT Enterprise Only
Intel®	Xeon®	E7-4850 v4	
Intel®	Xeon®	E7-8860 v3	IoT Enterprise Only
Intel®	Xeon®	E7-8860 v4	
Intel®	Xeon®	E7-8867 v3	IoT Enterprise Only
Intel®	Xeon®	E7-8867 v4	
Intel®	Xeon®	E7-8870 v3	IoT Enterprise Only
Intel®	Xeon®	E7-8870 v4	
Intel®	Xeon®	E7-8880 v3	IoT Enterprise Only
Intel®	Xeon®	E7-8880 v4	
Intel®	Xeon®	E7-8880L v3	IoT Enterprise Only
Intel®	Xeon®	E7-8890 v3	IoT Enterprise Only
Intel®	Xeon®	E7-8890 v4	
Intel®	Xeon®	E7-8891 v3	IoT Enterprise Only
Intel®	Xeon®	E7-8891 v4	
Intel®	Xeon®	E7-8893 v3	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E7-8893 v4	
Intel®	Xeon®	E7-8894 v4	
Intel®	Xeon®	Gold 5115	
Intel®	Xeon®	Gold 5118	
Intel®	Xeon®	Gold 5119T	
Intel®	Xeon®	Gold 5120	
Intel®	Xeon®	Gold 5120T	
Intel®	Xeon®	Gold 5122	
Intel®	Xeon®	Gold 6126	
Intel®	Xeon®	Gold 6126F	
Intel®	Xeon®	Gold 6126T	
Intel®	Xeon®	Gold 6128	
Intel®	Xeon®	Gold 6130	
Intel®	Xeon®	Gold 6130F	
Intel®	Xeon®	Gold 6130T	
Intel®	Xeon®	Gold 6132	
Intel®	Xeon®	Gold 6134	
Intel®	Xeon®	Gold 6136	
Intel®	Xeon®	Gold 6138	
Intel®	Xeon®	Gold 6138F	
Intel®	Xeon®	Gold 6138P	
Intel®	Xeon®	Gold 6138T	
Intel®	Xeon®	Gold 6140	
Intel®	Xeon®	Gold 6142	
Intel®	Xeon®	Gold 6142F	
Intel®	Xeon®	Gold 6144	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6146	
Intel®	Xeon®	Gold 6148	
Intel®	Xeon®	Gold 6148F	
Intel®	Xeon®	Gold 6150	
Intel®	Xeon®	Gold 6152	
Intel®	Xeon®	Gold 6154	
Intel®	Xeon®	Platinum 8153	
Intel®	Xeon®	Platinum 8156	
Intel®	Xeon®	Platinum 8158	
Intel®	Xeon®	Platinum 8160	
Intel®	Xeon®	Platinum 8160F	
Intel®	Xeon®	Platinum 8160T	
Intel®	Xeon®	Platinum 8164	
Intel®	Xeon®	Platinum 8168	
Intel®	Xeon®	Platinum 8170	
Intel®	Xeon®	Platinum 8176	
Intel®	Xeon®	Platinum 8176F	
Intel®	Xeon®	Platinum 8180	
Intel®	Xeon®	Silver 4108	
Intel®	Xeon®	Silver 4109T	
Intel®	Xeon®	Silver 4110	
Intel®	Xeon®	Silver 4112	
Intel®	Xeon®	Silver 4114	
Intel®	Xeon®	Silver 4114T	
Intel®	Xeon®	Silver 4116	
Intel®	Xeon®	Silver 4116T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-2123	
Intel®	Xeon®	W-2125	
Intel®	Xeon®	W-2133	
Intel®	Xeon®	W-2135	
Intel®	Xeon®	W-2145	
Intel®	Xeon®	W-2155	
Intel®	Xeon®	W-2175	
Intel®	Xeon®	W-2195	
Intel®	Xeon®	W-3175X	

Windows 10 LTSC 1809 Supported AMD Processors

Article • 11/10/2021

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicablility
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics	
AMD	A-Series	A10-6700 with Radeon™ HD 8670D	
AMD	A-Series	A10-6700T with Radeon™ HD 8650D	
AMD	A-Series	A10-6790B with Radeon™ HD 8670D	
AMD	A-Series	A10-6790K with Radeon™ HD 8670D	
AMD	A-Series	A10-6800B with Radeon™ HD 8670D	
AMD	A-Series	A10-6800K with Radeon™ HD 8670D	
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics	
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics	
AMD	A-Series	A10-7700K with Radeon™ R7 Series	
AMD	A-Series	A10-7800 with Radeon™ R7 Series	
AMD	A-Series	A10-7850K with Radeon™ R7 Series	
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution	
AMD	A-Series	A10-7860K with Radeon™ R7 Series	
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution	
AMD	A-Series	A10-7870K with Radeon™ R7 Series	

Manufacturer	Brand	Model	IoT Applicablility
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler	
AMD	A-Series	A10-8700P APU	
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics	
AMD	A-Series	A10-9600P APU	
AMD	A-Series	A10-9630P APU	
AMD	A-Series	A10-9700 APU	
AMD	A-Series	A10-9700E APU	
AMD	A-Series	A12-9700P APU	
AMD	A-Series	A12-9730P APU	
AMD	A-Series	A12-9800 APU	
AMD	A-Series	A12-9800E APU	
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics	
AMD	A-Series	A4-5000 with Radeon™ HD 8330	
AMD	A-Series	A4-5100 with Radeon™ HD 8330	
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics	
AMD	A-Series	A4-6300 with Radeon™ HD 8370D	
AMD	A-Series	A4-6320 with Radeon™ HD 8370D	
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics	
AMD	A-Series	A4-7300 with Radeon™ HD 8470D	
AMD	A-Series	A4-9120 APU	
AMD	A-Series	A4-9120C APU	
AMD	A-Series	A4-9125 APU	
AMD	A-Series	A6-5200 with Radeon™ HD 8400	
AMD	A-Series	A6-5200M with Radeon™ HD 8400	

Manufacturer	Brand	Model	IoT Applicablility
AMD	A-Series	A6-5350M with Radeon™ HD 8450G	
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics	
AMD	A-Series	A6-6400B with Radeon™ HD 8470D	
AMD	A-Series	A6-6400K with Radeon™ HD 8470D	
AMD	A-Series	A6-6420B with Radeon™ HD 8470D	
AMD	A-Series	A6-6420K with Radeon™ HD 8470D	
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics	
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics	
AMD	A-Series	A6-7400K with Radeon™ R5 Series	
AMD	A-Series	A6-7470K with Radeon™ R5 Series	
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics	
AMD	A-Series	A6-9210 APU	
AMD	A-Series	A6-9220 APU	
AMD	A-Series	A6-9220C APU	
AMD	A-Series	A6-9225 APU	
AMD	A-Series	A6-9500 APU	
AMD	A-Series	A6-9500E APU	
AMD	A-Series	A6-9550 APU	
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics	
AMD	A-Series	A8-6500 with Radeon™ HD 8570D	
AMD	A-Series	A8-6500B with Radeon™ HD 8570D	
AMD	A-Series	A8-6500T with Radeon™ HD 8550D	
AMD	A-Series	A8-6600K with Radeon™ HD 8570D	
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics	
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics	
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics	

Manufacturer	Brand	Model	IoT Applicablility
AMD	A-Series	A8-7600 with Radeon™ R7 Series	
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution	
AMD	A-Series	A8-7650K with Radeon™ R7 Series	
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution	
AMD	A-Series	A8-8600P APU	
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics	
AMD	A-Series	A8-9600 APU	
AMD	A-Series	A9-9410 APU	
AMD	A-Series	A9-9420 APU	
AMD	A-Series	A9-9425 APU	
AMD	Athlon™	200GE	
AMD	Athlon™	220GE	
AMD	Athlon™	240GE	
AMD	Athlon™	3000G	
AMD	Athlon™	300GE	
AMD	Athlon™	300U	
AMD	Athlon™	320GE	
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series	
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series	
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series	
AMD	Athlon™	Gold 3150C	
AMD	Athlon™	Gold 3150G (OEM Only)	
AMD	Athlon™	Gold 3150GE (OEM Only)	

Manufacturer	Brand	Model	IoT Applicablility
AMD	Athlon™	Gold 3150U	
AMD	Athlon™	Silver 3050C	
AMD	Athlon™	Silver 3050e	
AMD	Athlon™	Silver 3050GE (OEM Only)	
AMD	Athlon™	Silver 3050U	
AMD	Athlon™	X4 940	
AMD	Athlon™	X4 950	
AMD	Athlon™	X4 970	
AMD	EPYC™	7232P	
AMD	EPYC™	7251	
AMD	EPYC™	7252	
AMD	EPYC™	7261	
AMD	EPYC™	7262	
AMD	EPYC™	7272	
AMD	EPYC™	7281	
AMD	EPYC™	7282	
AMD	EPYC™	72F3	
AMD	EPYC™	7301	
AMD	EPYC™	7302	
AMD	EPYC™	7302P	
AMD	EPYC™	7313	
AMD	EPYC™	7313P	
AMD	EPYC™	7343	
AMD	EPYC™	7351	
AMD	EPYC™	7351P	
AMD	EPYC™	7352	

Manufacturer	Brand	Model	IoT Applicablility
AMD	EPYC™	7371	
AMD	EPYC™	73F3	
AMD	EPYC™	7401	
AMD	EPYC™	7401P	
AMD	EPYC™	7402	
AMD	EPYC™	7402P	
AMD	EPYC™	7413	
AMD	EPYC™	7443	
AMD	EPYC™	7443P	
AMD	EPYC™	7451	
AMD	EPYC™	7452	
AMD	EPYC™	7453	
AMD	EPYC™	74F3	
AMD	EPYC™	7501	
AMD	EPYC™	7502	
AMD	EPYC™	7502P	
AMD	EPYC™	7513	
AMD	EPYC™	7532	
AMD	EPYC™	7542	
AMD	EPYC™	7543	
AMD	EPYC™	7543P	
AMD	EPYC™	7551	
AMD	EPYC™	7551P	
AMD	EPYC™	7552	
AMD	EPYC™	75F3	
AMD	EPYC™	7601	

Manufacturer	Brand	Model	IoT Applicablility
AMD	EPYC™	7642	
AMD	EPYC™	7643	
AMD	EPYC™	7662	
AMD	EPYC™	7663	
AMD	EPYC™	7702	
AMD	EPYC™	7702P	
AMD	EPYC™	7713	
AMD	EPYC™	7713P	
AMD	EPYC™	7742	
AMD	EPYC™	7763	
AMD	EPYC™	7F32	
AMD	EPYC™	7F52	
AMD	EPYC™	7F72	
AMD	EPYC™	7H12	
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics	
AMD	E-Series	E1-2100 with Radeon™ HD 8210	
AMD	E-Series	E1-2200 with Radeon™ HD 8210	
AMD	E-Series	E1-2500 with Radeon™ HD 8240	
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics	
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics	
AMD	E-Series	E2-3000 with Radeon™ HD 8280	
AMD	E-Series	E2-3800 with Radeon™ HD 8280	
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics	
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics	
AMD	E-Series	E2-9010 APU	

Manufacturer	Brand	Model	IoT Applicablility
AMD	FX-Series	9800P APU	
AMD	FX-Series	9830P APU	
AMD	FX-Series	FX 6100	
AMD	FX-Series	FX-4100	
AMD	FX-Series	FX-4130	
AMD	FX-Series	FX-4170	
AMD	FX-Series	FX-4300	
AMD	FX-Series	FX-4320	
AMD	FX-Series	FX-4350	
AMD	FX-Series	FX-6200	
AMD	FX-Series	FX-6300	
AMD	FX-Series	FX-6350	
AMD	FX-Series	FX-6350 with Wraith cooler	
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics	
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics	
AMD	FX-Series	FX-8120	
AMD	FX-Series	FX-8150	
AMD	FX-Series	FX-8300	
AMD	FX-Series	FX-8310	
AMD	FX-Series	FX-8320	
AMD	FX-Series	FX-8320E	
AMD	FX-Series	FX-8350	
AMD	FX-Series	FX-8350 with Wraith cooler	
AMD	FX-Series	FX-8370	
AMD	FX-Series	FX-8370 with Wraith cooler	
AMD	FX-Series	FX-8370E	

Manufacturer	Brand	Model	IoT Applicablility
AMD	FX-Series	FX-8800P APU	
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics	
AMD	FX-Series	FX-9370	
AMD	FX-Series	FX-9590	
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics	
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics	
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics	
AMD	PRO A-Series	A10-8700B APU	
AMD	PRO A-Series	A10-8730B APU	
AMD	PRO A-Series	A10-8750B APU	
AMD	PRO A-Series	A10-8770 APU	
AMD	PRO A-Series	A10-8770E APU	
AMD	PRO A-Series	A10-8850B APU	
AMD	PRO A-Series	A10-9700 APU	
AMD	PRO A-Series	A10-9700B APU	
AMD	PRO A-Series	A10-9700E APU	
AMD	PRO A-Series	A10-9730B APU	
AMD	PRO A-Series	A12-8800B APU	
AMD	PRO A-Series	A12-8830B APU	
AMD	PRO A-Series	A12-8870 APU	
AMD	PRO A-Series	A12-8870E APU	
AMD	PRO A-Series	A12-9800 APU	
AMD	PRO A-Series	A12-9800B APU	
AMD	PRO A-Series	A12-9800E APU	

Manufacturer	Brand	Model	IoT Applicablility
AMD	PRO A-Series	A12-9830B APU	
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics	
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics	
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D	
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics	
AMD	PRO A-Series	A4-4350B APU	
AMD	PRO A-Series	A4-5350B APU	
AMD	PRO A-Series	A4-8350B APU	
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics	
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics	
AMD	PRO A-Series	A6-7350B APU	
AMD	PRO A-Series	A6-8350B APU	
AMD	PRO A-Series	A6-8500B APU	
AMD	PRO A-Series	A6-8530B APU	
AMD	PRO A-Series	A6-8550B APU	
AMD	PRO A-Series	A6-8570 APU	
AMD	PRO A-Series	A6-8570E APU	
AMD	PRO A-Series	A6-9500 APU	
AMD	PRO A-Series	A6-9500B APU	
AMD	PRO A-Series	A6-9500E APU	
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics	
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics	

Manufacturer	Brand	Model	IoT Applicablility
AMD	PRO A-Series	A8-8600B APU	
AMD	PRO A-Series	A8-8650B APU	
AMD	PRO A-Series	A8-9600 APU	
AMD	PRO A-Series	A8-9600B APU	
AMD	PRO A-Series	A8-9630B	
AMD	Ryzen™ 3	1200	
AMD	Ryzen™ 3	1300X	
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics	
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics	
AMD	Ryzen™ 3	2200U	
AMD	Ryzen™ 3	2300U	
AMD	Ryzen™ 3	2300X	
AMD	Ryzen™ 3 PRO	1200	
AMD	Ryzen™ 3 PRO	1300	
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics	
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics	
AMD	Ryzen™ 3 PRO	2300U	
AMD	Ryzen™ 5	1400	
AMD	Ryzen™ 5	1500X	
AMD	Ryzen™ 5	1600	
AMD	Ryzen™ 5	1600 (AF)	
AMD	Ryzen™ 5	1600X	
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics	
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics	

Manufacturer	Brand	Model	IoT Applicablility
AMD	Ryzen™ 5	2500U	
AMD	Ryzen™ 5	2500X	
AMD	Ryzen™ 5	2600	
AMD	Ryzen™ 5	2600E	
AMD	Ryzen™ 5	2600H	
AMD	Ryzen™ 5	2600X	
AMD	Ryzen™ 5 PRO	1500	
AMD	Ryzen™ 5 PRO	1600	
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics	
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics	
AMD	Ryzen™ 5 PRO	2500U	
AMD	Ryzen™ 5 PRO	2600	
AMD	Ryzen™ 7	1700 Processor	
AMD	Ryzen™ 7	1700X	
AMD	Ryzen™ 7	1800X	
AMD	Ryzen™ 7	2700	
AMD	Ryzen™ 7	2700E Processor	
AMD	Ryzen™ 7	2700U	
AMD	Ryzen™ 7	2700X	
AMD	Ryzen™ 7	2800H	
AMD	Ryzen™ 7 PRO	1700	
AMD	Ryzen™ 7 PRO	1700X Processor	
AMD	Ryzen™ 7 PRO	2700	
AMD	Ryzen™ 7 PRO	2700U	
AMD	Ryzen™ 7 PRO	2700X	

Manufacturer	Brand	Model	IoT Applicablility
AMD	Ryzen™ Threadripper™	1900X	
AMD	Ryzen™ Threadripper™	1920X	
AMD	Ryzen™ Threadripper™	1950X	
AMD	Ryzen™ Threadripper™	2920X	
AMD	Ryzen™ Threadripper™	2950X	
AMD	Ryzen™ Threadripper™	2970WX	
AMD	Ryzen™ Threadripper™	2990WX	

Windows 10 LTSC 1809 Supported Intel Processors

Article • 07/15/2022

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Atom®	E3805	IoT Enterprise Only
Intel®	Atom®	E3827	IoT Enterprise Only
Intel®	Atom®	E3826	IoT Enterprise Only
Intel®	Atom®	E3815	IoT Enterprise Only
Intel®	Atom®	E3845	IoT Enterprise Only
Intel®	Atom®	E3825	IoT Enterprise Only
Intel®	Atom®	x5-E3930	IoT Enterprise Only
Intel®	Atom®	x5-E3940	IoT Enterprise Only
Intel®	Atom®	x7-E3950	IoT Enterprise Only
Intel®	Atom®	x5-E8000	IoT Enterprise Only
Intel®	Celeron®	3205U	
Intel®	Celeron®	3215U	
Intel®	Celeron®	3755U	
Intel®	Celeron®	3765U	
Intel®	Celeron®	3855U	
Intel®	Celeron®	3865U	
Intel®	Celeron®	3867U	
Intel®	Celeron®	3955U	
Intel®	Celeron®	3965U	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Celeron®	3965Y	
Intel®	Celeron®	4205U	
Intel®	Celeron®	4305U	
Intel®	Celeron®	4305UE	
Intel®	Celeron®	5205U	
Intel®	Celeron®	5305U	
Intel®	Celeron®	6305	IoT Enterprise Only
Intel®	Celeron®	6305E	IoT Enterprise Only
Intel®	Celeron®	G3900	
Intel®	Celeron®	G3900E	
Intel®	Celeron®	G3900T	
Intel®	Celeron®	G3900TE	
Intel®	Celeron®	G3902E	
Intel®	Celeron®	G3920	
Intel®	Celeron®	G3930	
Intel®	Celeron®	G3930E	
Intel®	Celeron®	G3930T	
Intel®	Celeron®	G3930TE	
Intel®	Celeron®	G3950	
Intel®	Celeron®	G4900	
Intel®	Celeron®	G4900T	
Intel®	Celeron®	G4920	
Intel®	Celeron®	G4930	
Intel®	Celeron®	G4930E	
Intel®	Celeron®	G4930T	
Intel®	Celeron®	G4932E	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Celeron®	G4950	
Intel®	Celeron®	G5900	
Intel®	Celeron®	G5900E	
Intel®	Celeron®	G5900T	
Intel®	Celeron®	G5900TE	
Intel®	Celeron®	G5905	
Intel®	Celeron®	G5905T	
Intel®	Celeron®	G5920	
Intel®	Celeron®	G5925	
Intel®	Celeron®	J1900	IoT Enterprise Only
Intel®	Celeron®	J3355	IoT Enterprise Only
Intel®	Celeron®	J3355E	IoT Enterprise Only
Intel®	Celeron®	J3455	IoT Enterprise Only
Intel®	Celeron®	J3455E	IoT Enterprise Only
Intel®	Celeron®	J4005	
Intel®	Celeron®	J4025	
Intel®	Celeron®	J4105	
Intel®	Celeron®	J4125	
Intel®	Celeron®	N2807	IoT Enterprise Only
Intel®	Celeron®	N2920	IoT Enterprise Only
Intel®	Celeron®	N2930	IoT Enterprise Only
Intel®	Celeron®	N3010	IoT Enterprise Only
Intel®	Celeron®	N3060	IoT Enterprise Only
Intel®	Celeron®	N3160	IoT Enterprise Only
Intel®	Celeron®	N3350	IoT Enterprise Only
Intel®	Celeron®	N4000	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Celeron®	N4020	
Intel®	Celeron®	N4100	
Intel®	Celeron®	N4120	
Intel®	Core™	i3-1000G1	
Intel®	Core™	i3-1000G4	
Intel®	Core™	i3-1005G1	
Intel®	Core™	i3-10100	
Intel®	Core™	i3-10100E	
Intel®	Core™	i3-10100F	
Intel®	Core™	i3-10100T	
Intel®	Core™	i3-10100TE	
Intel®	Core™	i3-10100Y	
Intel®	Core™	i3-10105	
Intel®	Core™	i3-10105F	
Intel®	Core™	i3-10105T	
Intel®	Core™	i3-10110U	
Intel®	Core™	i3-10110Y	
Intel®	Core™	i3-10300	
Intel®	Core™	i3-10300T	
Intel®	Core™	i3-10305	
Intel®	Core™	i3-10305T	
Intel®	Core™	i3-10320	
Intel®	Core™	i3-10325	
Intel®	Core™	i3-1110G4	IoT Enterprise Only
Intel®	Core™	i3-1115G4	IoT Enterprise Only
Intel®	Core™	i3-1115G4E	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i3-1115GRE	IoT Enterprise Only
Intel®	Core™	i3-1120G4	IoT Enterprise Only
Intel®	Core™	i3-1125G4	IoT Enterprise Only
Intel®	Core™	i3-5005U	
Intel®	Core™	i3-5010U	
Intel®	Core™	i3-5015U	
Intel®	Core™	i3-5020U	
Intel®	Core™	i3-5157U	
Intel®	Core™	i3-6006U	
Intel®	Core™	i3-6098P	
Intel®	Core™	i3-6100	
Intel®	Core™	i3-6100E	
Intel®	Core™	i3-6100H	
Intel®	Core™	i3-6100T	
Intel®	Core™	i3-6100TE	
Intel®	Core™	i3-6100U	
Intel®	Core™	i3-6102E	
Intel®	Core™	i3-6157U	
Intel®	Core™	i3-6167U	
Intel®	Core™	i3-6300	
Intel®	Core™	i3-6300T	
Intel®	Core™	i3-6320	
Intel®	Core™	i3-7020U	
Intel®	Core™	i3-7100	
Intel®	Core™	i3-7100E	
Intel®	Core™	i3-7100H	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i3-7100T	
Intel®	Core™	i3-7100U	
Intel®	Core™	i3-7101E	
Intel®	Core™	i3-7101TE	
Intel®	Core™	i3-7102E	
Intel®	Core™	i3-7130U	
Intel®	Core™	i3-7167U	
Intel®	Core™	i3-7300	
Intel®	Core™	i3-7300T	
Intel®	Core™	i3-7320	
Intel®	Core™	i3-7350K	
Intel®	Core™	i3-8100	
Intel®	Core™	i3-8100B	
Intel®	Core™	i3-8100H	
Intel®	Core™	i3-8100T	
Intel®	Core™	i3-8109U	
Intel®	Core™	i3-8130U	
Intel®	Core™	i3-8140U	
Intel®	Core™	i3-8145U	
Intel®	Core™	i3-8145UE	
Intel®	Core™	i3-8300	
Intel®	Core™	i3-8300T	
Intel®	Core™	i3-8350K	
Intel®	Core™	i3-9100	
Intel®	Core™	i3-9100E	
Intel®	Core™	i3-9100F	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i3-9100HL	
Intel®	Core™	i3-9100T	
Intel®	Core™	i3-9100TE	
Intel®	Core™	i3-9300	
Intel®	Core™	i3-9300T	
Intel®	Core™	i3-9320	
Intel®	Core™	i3-9350K	
Intel®	Core™	i3-9350KF	
Intel®	Core™	i5-10200H	
Intel®	Core™	i5-10210U	
Intel®	Core™	i5-10210Y	
Intel®	Core™	i5-10300H	
Intel®	Core™	i5-1030G4	
Intel®	Core™	i5-1030G7	
Intel®	Core™	i5-10310U	
Intel®	Core™	i5-10310Y	
Intel®	Core™	i5-1035G1	
Intel®	Core™	i5-1035G4	
Intel®	Core™	i5-1035G7	
Intel®	Core™	i5-1038NG7	
Intel®	Core™	i5-10400	
Intel®	Core™	i5-10400F	
Intel®	Core™	i5-10400H	
Intel®	Core™	i5-10400T	
Intel®	Core™	i5-10500	
Intel®	Core™	i5-10500E	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i5-10500H	
Intel®	Core™	i5-10500T	
Intel®	Core™	i5-10500TE	
Intel®	Core™	i5-10600	
Intel®	Core™	i5-10600K	
Intel®	Core™	i5-10600KF	
Intel®	Core™	i5-10600T	
Intel®	Core™	i5-11300H	IoT Enterprise Only
Intel®	Core™	i5-1130G7	IoT Enterprise Only
Intel®	Core™	i5-1135G7	IoT Enterprise Only
Intel®	Core™	i5-1140G7	IoT Enterprise Only
Intel®	Core™	i5-1145G7	IoT Enterprise Only
Intel®	Core™	i5-1145G7E	IoT Enterprise Only
Intel®	Core™	i5-1145GRE	IoT Enterprise Only
Intel®	Core™	i5-5200U	
Intel®	Core™	i5-5250U	
Intel®	Core™	i5-5257U	
Intel®	Core™	i5-5287U	
Intel®	Core™	i5-5300U	
Intel®	Core™	i5-5350H	
Intel®	Core™	i5-5350U	
Intel®	Core™	i5-5575R	
Intel®	Core™	i5-5675C	
Intel®	Core™	i5-5675R	
Intel®	Core™	i5-6200U	
Intel®	Core™	i5-6260U	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i5-6267U	
Intel®	Core™	i5-6287U	
Intel®	Core™	i5-6300HQ	
Intel®	Core™	i5-6300U	
Intel®	Core™	i5-6350HQ	
Intel®	Core™	i5-6360U	
Intel®	Core™	i5-6400	
Intel®	Core™	i5-6400T	
Intel®	Core™	i5-6402P	
Intel®	Core™	i5-6440EQ	
Intel®	Core™	i5-6440HQ	
Intel®	Core™	i5-6442EQ	
Intel®	Core™	i5-6500	
Intel®	Core™	i5-6500T	
Intel®	Core™	i5-6500TE	
Intel®	Core™	i5-6585R	
Intel®	Core™	i5-6600	
Intel®	Core™	i5-6600K	
Intel®	Core™	i5-6600T	
Intel®	Core™	i5-6685R	
Intel®	Core™	i5-7200U	
Intel®	Core™	i5-7260U	
Intel®	Core™	i5-7267U	
Intel®	Core™	i5-7287U	
Intel®	Core™	i5-7300HQ	
Intel®	Core™	i5-7300U	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i5-7360U	
Intel®	Core™	i5-7400	
Intel®	Core™	i5-7400T	
Intel®	Core™	i5-7440EQ	
Intel®	Core™	i5-7440HQ	
Intel®	Core™	i5-7442EQ	
Intel®	Core™	i5-7500	
Intel®	Core™	i5-7500T	
Intel®	Core™	i5-7600	
Intel®	Core™	i5-7600K	
Intel®	Core™	i5-7600T	
Intel®	Core™	i5-7640X	
Intel®	Core™	i5-7Y54	
Intel®	Core™	i5-7Y57	
Intel®	Core™	i5-8200Y	
Intel®	Core™	i5-8210Y	
Intel®	Core™	i5-8250U	
Intel®	Core™	i5-8257U	
Intel®	Core™	i5-8259U	
Intel®	Core™	i5-8260U	
Intel®	Core™	i5-8265U	
Intel®	Core™	i5-8269U	
Intel®	Core™	i5-8279U	
Intel®	Core™	i5-8300H	
Intel®	Core™	i5-8310Y	
Intel®	Core™	i5-8350U	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i5-8365U	
Intel®	Core™	i5-8365UE	
Intel®	Core™	i5-8400	
Intel®	Core™	i5-8400	
Intel®	Core™	i5-8400B	
Intel®	Core™	i5-8400H	
Intel®	Core™	i5-8400T	
Intel®	Core™	i5-8500	
Intel®	Core™	i5-8500	
Intel®	Core™	i5-8500B	
Intel®	Core™	i5-8500T	
Intel®	Core™	i5-8600	
Intel®	Core™	i5-8600K	
Intel®	Core™	i5-8600T	
Intel®	Core™	i5-9300H	
Intel®	Core™	i5-9300HF	
Intel®	Core™	i5-9400	
Intel®	Core™	i5-9400F	
Intel®	Core™	i5-9400H	
Intel®	Core™	i5-9400T	
Intel®	Core™	i5-9500	
Intel®	Core™	i5-9500E	
Intel®	Core™	i5-9500F	
Intel®	Core™	i5-9500T	
Intel®	Core™	i5-9500TE	
Intel®	Core™	i5-9600	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i5-9600K	
Intel®	Core™	i5-9600KF	
Intel®	Core™	i5-9600T	
Intel®	Core™	i7-10510U	
Intel®	Core™	i7-10510Y	
Intel®	Core™	i7-1060G7	
Intel®	Core™	i7-10610U	
Intel®	Core™	i7-1065G7	
Intel®	Core™	i7-1068NG7	
Intel®	Core™	i7-10700	
Intel®	Core™	i7-10700E	
Intel®	Core™	i7-10700F	
Intel®	Core™	i7-10700K	
Intel®	Core™	i7-10700KF	
Intel®	Core™	i7-10700T	
Intel®	Core™	i7-10700TE	
Intel®	Core™	i7-10710U	
Intel®	Core™	i7-10750H	
Intel®	Core™	i7-10810U	
Intel®	Core™	i7-10850H	
Intel®	Core™	i7-10870H	
Intel®	Core™	i7-10875H	
Intel®	Core™	i7-11370H	IoT Enterprise Only
Intel®	Core™	i7-11375H	IoT Enterprise Only
Intel®	Core™	i7-1160G7	IoT Enterprise Only
Intel®	Core™	i7-1165G7	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i7-1165G7	IoT Enterprise Only
Intel®	Core™	i7-1180G7	IoT Enterprise Only
Intel®	Core™	i7-1185G7	IoT Enterprise Only
Intel®	Core™	i7-1185G7E	IoT Enterprise Only
Intel®	Core™	i7-1185GRE	IoT Enterprise Only
Intel®	Core™	i7-5500U	
Intel®	Core™	i7-5550U	
Intel®	Core™	i7-5557U	
Intel®	Core™	i7-5600U	
Intel®	Core™	i7-5650U	
Intel®	Core™	i7-5700EQ	
Intel®	Core™	i7-5700HQ	
Intel®	Core™	i7-5750HQ	
Intel®	Core™	i7-5775C	
Intel®	Core™	i7-5775R	
Intel®	Core™	i7-5850EQ	
Intel®	Core™	i7-5850HQ	
Intel®	Core™	i7-5950HQ	
Intel®	Core™	i7-6500U	
Intel®	Core™	i7-6560U	
Intel®	Core™	i7-6567U	
Intel®	Core™	i7-6600U	
Intel®	Core™	i7-6650U	
Intel®	Core™	i7-6660U	
Intel®	Core™	i7-6700	
Intel®	Core™	i7-6700HQ	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i7-6700K	
Intel®	Core™	i7-6700T	
Intel®	Core™	i7-6700TE	
Intel®	Core™	i7-6770HQ	
Intel®	Core™	i7-6785R	
Intel®	Core™	i7-6820EQ	
Intel®	Core™	i7-6820HK	
Intel®	Core™	i7-6820HQ	
Intel®	Core™	i7-6822EQ	
Intel®	Core™	i7-6870HQ	
Intel®	Core™	i7-6920HQ	
Intel®	Core™	i7-6970HQ	
Intel®	Core™	i7-7500U	
Intel®	Core™	i7-7560U	
Intel®	Core™	i7-7567U	
Intel®	Core™	i7-7600U	
Intel®	Core™	i7-7660U	
Intel®	Core™	i7-7700	
Intel®	Core™	i7-7700HQ	
Intel®	Core™	i7-7700K	
Intel®	Core™	i7-7700T	
Intel®	Core™	i7-7740X	
Intel®	Core™	i7-7800X	
Intel®	Core™	i7-7820EQ	
Intel®	Core™	i7-7820HK	
Intel®	Core™	i7-7820HQ	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i7-7820X	
Intel®	Core™	i7-7920HQ	
Intel®	Core™	i7-7Y75	
Intel®	Core™	i7-8086K	
Intel®	Core™	i7-8500Y	
Intel®	Core™	i7-8550U	
Intel®	Core™	i7-8557U	
Intel®	Core™	i7-8559U	
Intel®	Core™	i7-8565U	
Intel®	Core™	i7-8569U	
Intel®	Core™	i7-8650U	
Intel®	Core™	i7-8665U	
Intel®	Core™	i7-8665UE	
Intel®	Core™	i7-8700	
Intel®	Core™	i7-8700	
Intel®	Core™	i7-8700B	
Intel®	Core™	i7-8700K	
Intel®	Core™	i7-8700T	
Intel®	Core™	i7-8750H	
Intel®	Core™	i7-8850H	
Intel®	Core™	i7-9700	
Intel®	Core™	i7-9700E	
Intel®	Core™	i7-9700F	
Intel®	Core™	i7-9700K	
Intel®	Core™	i7-9700KF	
Intel®	Core™	i7-9700T	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Core™	i7-9700TE	
Intel®	Core™	i7-9750H	
Intel®	Core™	i7-9750HF	
Intel®	Core™	i7-9800X	
Intel®	Core™	i7-9850H	
Intel®	Core™	i7-9850HE	
Intel®	Core™	i7-9850HL	
Intel®	Pentium®	N3710	IoT Enterprise Only
Intel®	Pentium®	N4200	IoT Enterprise Only
Intel®	Pentium®	3805U	
Intel®	Pentium®	3825U	
Intel®	Pentium®	4405U	
Intel®	Pentium®	4405Y	
Intel®	Pentium®	6805	
Intel®	Pentium®	D1507	
Intel®	Pentium®	D1508	
Intel®	Pentium®	D1509	
Intel®	Pentium®	D1517	
Intel®	Pentium®	D1519	
Intel®	Pentium®	G4400	
Intel®	Pentium®	G4400T	
Intel®	Pentium®	G4400TE	
Intel®	Pentium®	G4500	
Intel®	Pentium®	G4500T	
Intel®	Pentium®	G4520	
Intel®	Pentium®	G4560	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Pentium®	G4560T	
Intel®	Pentium®	G4600	
Intel®	Pentium®	G4600T	
Intel®	Pentium®	G4620	
Intel®	Pentium®	Gold 4410Y	
Intel®	Pentium®	Gold 4415U	
Intel®	Pentium®	Gold 4415Y	
Intel®	Pentium®	Gold 4417U	
Intel®	Pentium®	Gold 4425Y	
Intel®	Pentium®	Gold 5405U	
Intel®	Pentium®	Gold 6405U	
Intel®	Pentium®	Gold 6500Y	
Intel®	Pentium®	Gold G5400	
Intel®	Pentium®	Gold G5400T	
Intel®	Pentium®	Gold G5420	
Intel®	Pentium®	Gold G5420T	
Intel®	Pentium®	Gold G5500	
Intel®	Pentium®	Gold G5500T	
Intel®	Pentium®	Gold G5600	
Intel®	Pentium®	Gold G5600T	
Intel®	Pentium®	Gold G5620	
Intel®	Pentium®	Gold G6400	
Intel®	Pentium®	Gold G6400E	
Intel®	Pentium®	Gold G6400T	
Intel®	Pentium®	Gold G6400TE	
Intel®	Pentium®	Gold G6405	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Pentium®	Gold G6405T	
Intel®	Pentium®	Gold G6500	
Intel®	Pentium®	Gold G6500T	
Intel®	Pentium®	Gold G6505	
Intel®	Pentium®	Gold G6505T	
Intel®	Pentium®	Gold G6600	
Intel®	Pentium®	Gold G6605	
Intel®	Pentium®	Silver J5005	
Intel®	Pentium®	Silver J5040	
Intel®	Pentium®	Silver N5000	
Intel®	Pentium®	Silver N5030	
Intel®	Xeon®	Bronze 3104	
Intel®	Xeon®	Bronze 3106	
Intel®	Xeon®	Bronze 3204	
Intel®	Xeon®	Bronze 3206R	
Intel®	Xeon®	D-1513N	
Intel®	Xeon®	D-1518	
Intel®	Xeon®	D-1520	
Intel®	Xeon®	D-1521	
Intel®	Xeon®	D-1523N	
Intel®	Xeon®	D-1527	
Intel®	Xeon®	D-1528	
Intel®	Xeon®	D-1529	
Intel®	Xeon®	D-1531	
Intel®	Xeon®	D-1533N	
Intel®	Xeon®	D-1537	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	D-1539	
Intel®	Xeon®	D-1540	
Intel®	Xeon®	D-1541	
Intel®	Xeon®	D-1543N	
Intel®	Xeon®	D-1548	
Intel®	Xeon®	D-1553N	
Intel®	Xeon®	D-1557	
Intel®	Xeon®	D-1559	
Intel®	Xeon®	D-1567	
Intel®	Xeon®	D-1571	
Intel®	Xeon®	D-1577	
Intel®	Xeon®	D-2123IT	
Intel®	Xeon®	D-2141I	
Intel®	Xeon®	D-2142IT	
Intel®	Xeon®	D-2143IT	
Intel®	Xeon®	D-2145NT	
Intel®	Xeon®	D-2146NT	
Intel®	Xeon®	D-2161I	
Intel®	Xeon®	D-2163IT	
Intel®	Xeon®	D-2166NT	
Intel®	Xeon®	D-2173IT	
Intel®	Xeon®	D-2177NT	
Intel®	Xeon®	D-2183IT	
Intel®	Xeon®	D-2187NT	
Intel®	Xeon®	E-2124	
Intel®	Xeon®	E-2124G	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	E-2126G	
Intel®	Xeon®	E-2134	
Intel®	Xeon®	E-2136	
Intel®	Xeon®	E-2144G	
Intel®	Xeon®	E-2146G	
Intel®	Xeon®	E-2174G	
Intel®	Xeon®	E-2176G	
Intel®	Xeon®	E-2176M	
Intel®	Xeon®	E-2186G	
Intel®	Xeon®	E-2186M	
Intel®	Xeon®	E-2224	
Intel®	Xeon®	E-2224G	
Intel®	Xeon®	E-2226G	
Intel®	Xeon®	E-2226GE	
Intel®	Xeon®	E-2234	
Intel®	Xeon®	E-2236	
Intel®	Xeon®	E-2244G	
Intel®	Xeon®	E-2246G	
Intel®	Xeon®	E-2254ME	
Intel®	Xeon®	E-2254ML	
Intel®	Xeon®	E-2274G	
Intel®	Xeon®	E-2276G	
Intel®	Xeon®	E-2276M	
Intel®	Xeon®	E-2276ME	
Intel®	Xeon®	E-2276ML	
Intel®	Xeon®	E-2278G	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	E-2278GE	
Intel®	Xeon®	E-2278GEL	
Intel®	Xeon®	E-2286G	
Intel®	Xeon®	E-2286M	
Intel®	Xeon®	E-2288G	
Intel®	Xeon®	E3-1220 v5	
Intel®	Xeon®	E3-1220 v6	
Intel®	Xeon®	E3-1225 v5	
Intel®	Xeon®	E3-1225 v6	
Intel®	Xeon®	E3-1230 v5	
Intel®	Xeon®	E3-1230 v6	
Intel®	Xeon®	E3-1235L v5	
Intel®	Xeon®	E3-1240 v5	
Intel®	Xeon®	E3-1240 v6	
Intel®	Xeon®	E3-1240L v5	
Intel®	Xeon®	E3-1245 v5	
Intel®	Xeon®	E3-1245 v6	
Intel®	Xeon®	E3-1258L v4	
Intel®	Xeon®	E3-1260L v5	
Intel®	Xeon®	E3-1265L v4	
Intel®	Xeon®	E3-1268L v5	
Intel®	Xeon®	E3-1270 v5	
Intel®	Xeon®	E3-1270 v6	
Intel®	Xeon®	E3-1275 v5	
Intel®	Xeon®	E3-1275 v6	
Intel®	Xeon®	E3-1278L v4	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	E3-1280 v5	
Intel®	Xeon®	E3-1280 v6	
Intel®	Xeon®	E3-1285 v4	
Intel®	Xeon®	E3-1285 v6	
Intel®	Xeon®	E3-1285L v4	
Intel®	Xeon®	E3-1501L v6	
Intel®	Xeon®	E3-1501M v6	
Intel®	Xeon®	E3-1505L v5	
Intel®	Xeon®	E3-1505L v6	
Intel®	Xeon®	E3-1505M v5	
Intel®	Xeon®	E3-1505M v6	
Intel®	Xeon®	E3-1515M v5	
Intel®	Xeon®	E3-1535M v5	
Intel®	Xeon®	E3-1535M v6	
Intel®	Xeon®	E3-1545M v5	
Intel®	Xeon®	E3-1558L v5	
Intel®	Xeon®	E3-1565L v5	
Intel®	Xeon®	E3-1575M v5	
Intel®	Xeon®	E3-1578L v5	
Intel®	Xeon®	E3-1585 v5	
Intel®	Xeon®	E3-1585L v5	
Intel®	Xeon®	E5-1620 v4	
Intel®	Xeon®	E5-1630 v4	
Intel®	Xeon®	E5-1650 v4	
Intel®	Xeon®	E5-1660 v4	
Intel®	Xeon®	E5-1680 v4	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	E5-2603 v4	
Intel®	Xeon®	E5-2608L v4	
Intel®	Xeon®	E5-2609 v4	
Intel®	Xeon®	E5-2618L v4	
Intel®	Xeon®	E5-2620 v4	
Intel®	Xeon®	E5-2623 v4	
Intel®	Xeon®	E5-2628L v4	
Intel®	Xeon®	E5-2630 v4	
Intel®	Xeon®	E5-2630L v4	
Intel®	Xeon®	E5-2637 v4	
Intel®	Xeon®	E5-2640 v4	
Intel®	Xeon®	E5-2643 v4	
Intel®	Xeon®	E5-2648L v4	
Intel®	Xeon®	E5-2650 v4	
Intel®	Xeon®	E5-2650L v4	
Intel®	Xeon®	E5-2658 v4	
Intel®	Xeon®	E5-2660 v4	
Intel®	Xeon®	E5-2667 v4	
Intel®	Xeon®	E5-2680 v4	
Intel®	Xeon®	E5-2683 v4	
Intel®	Xeon®	E5-2687W v4	
Intel®	Xeon®	E5-2690 v4	
Intel®	Xeon®	E5-2695 v4	
Intel®	Xeon®	E5-2697 v4	
Intel®	Xeon®	E5-2697A v4	
Intel®	Xeon®	E5-2698 v4	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	E5-2699 v4	
Intel®	Xeon®	E5-2699A v4	
Intel®	Xeon®	E5-2699R v4	
Intel®	Xeon®	E5-4610 v4	
Intel®	Xeon®	E5-4620 v4	
Intel®	Xeon®	E5-4627 v4	
Intel®	Xeon®	E5-4628L v4	
Intel®	Xeon®	E5-4640 v4	
Intel®	Xeon®	E5-4650 v4	
Intel®	Xeon®	E5-4655 v4	
Intel®	Xeon®	E5-4660 v4	
Intel®	Xeon®	E5-4667 v4	
Intel®	Xeon®	E5-4669 v4	
Intel®	Xeon®	E7-4809 v4	
Intel®	Xeon®	E7-4820 v4	
Intel®	Xeon®	E7-4830 v4	
Intel®	Xeon®	E7-4850 v4	
Intel®	Xeon®	E7-8860 v4	
Intel®	Xeon®	E7-8867 v4	
Intel®	Xeon®	E7-8870 v4	
Intel®	Xeon®	E7-8880 v4	
Intel®	Xeon®	E7-8890 v4	
Intel®	Xeon®	E7-8891 v4	
Intel®	Xeon®	E7-8893 v4	
Intel®	Xeon®	E7-8894 v4	
Intel®	Xeon®	Gold 5115	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Gold 5118	
Intel®	Xeon®	Gold 5119T	
Intel®	Xeon®	Gold 5120	
Intel®	Xeon®	Gold 5120T	
Intel®	Xeon®	Gold 5122	
Intel®	Xeon®	Gold 5215	
Intel®	Xeon®	Gold 5215L	
Intel®	Xeon®	Gold 5217	
Intel®	Xeon®	Gold 5218B	
Intel®	Xeon®	Gold 5218N	
Intel®	Xeon®	Gold 5218R	
Intel®	Xeon®	Gold 5218T	
Intel®	Xeon®	Gold 5220	
Intel®	Xeon®	Gold 5220R	
Intel®	Xeon®	Gold 5220S	
Intel®	Xeon®	Gold 5220T	
Intel®	Xeon®	Gold 5222	
Intel®	Xeon®	Gold 5315Y	
Intel®	Xeon®	Gold 5317	
Intel®	Xeon®	Gold 5318N	
Intel®	Xeon®	Gold 5318S	
Intel®	Xeon®	Gold 5320	
Intel®	Xeon®	Gold 5320T	
Intel®	Xeon®	Gold 6126	
Intel®	Xeon®	Gold 6126F	
Intel®	Xeon®	Gold 6126T	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Gold 6128	
Intel®	Xeon®	Gold 6130	
Intel®	Xeon®	Gold 6130F	
Intel®	Xeon®	Gold 6130T	
Intel®	Xeon®	Gold 6132	
Intel®	Xeon®	Gold 6134	
Intel®	Xeon®	Gold 6136	
Intel®	Xeon®	Gold 6138	
Intel®	Xeon®	Gold 6138F	
Intel®	Xeon®	Gold 6138P	
Intel®	Xeon®	Gold 6138T	
Intel®	Xeon®	Gold 6140	
Intel®	Xeon®	Gold 6142	
Intel®	Xeon®	Gold 6142F	
Intel®	Xeon®	Gold 6144	
Intel®	Xeon®	Gold 6146	
Intel®	Xeon®	Gold 6148	
Intel®	Xeon®	Gold 6148F	
Intel®	Xeon®	Gold 6150	
Intel®	Xeon®	Gold 6152	
Intel®	Xeon®	Gold 6154	
Intel®	Xeon®	Gold 6208U	
Intel®	Xeon®	Gold 6209U	
Intel®	Xeon®	Gold 6210U	
Intel®	Xeon®	Gold 6212U	
Intel®	Xeon®	Gold 6222V	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Gold 6226	
Intel®	Xeon®	Gold 6226R	
Intel®	Xeon®	Gold 6230	
Intel®	Xeon®	Gold 6230N	
Intel®	Xeon®	Gold 6230R	
Intel®	Xeon®	Gold 6230T	
Intel®	Xeon®	Gold 6238	
Intel®	Xeon®	Gold 6238L	
Intel®	Xeon®	Gold 6238T	
Intel®	Xeon®	Gold 6240	
Intel®	Xeon®	Gold 6240L	
Intel®	Xeon®	Gold 6240R	
Intel®	Xeon®	Gold 6240Y	
Intel®	Xeon®	Gold 6242	
Intel®	Xeon®	Gold 6242R	
Intel®	Xeon®	Gold 6244	
Intel®	Xeon®	Gold 6246R	
Intel®	Xeon®	Gold 6248	
Intel®	Xeon®	Gold 6248R	
Intel®	Xeon®	Gold 6250	
Intel®	Xeon®	Gold 6250L	
Intel®	Xeon®	Gold 6252	
Intel®	Xeon®	Gold 6252N	
Intel®	Xeon®	Gold 6254	
Intel®	Xeon®	Gold 6256	
Intel®	Xeon®	Gold 6258R	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Gold 6262V	
Intel®	Xeon®	Gold 6312U	
Intel®	Xeon®	Gold 6314U	
Intel®	Xeon®	Gold 6326	
Intel®	Xeon®	Gold 6330	
Intel®	Xeon®	Gold 6330N	
Intel®	Xeon®	Gold 6334	
Intel®	Xeon®	Gold 6336Y	
Intel®	Xeon®	Gold 6338	
Intel®	Xeon®	Gold 6338N	
Intel®	Xeon®	Gold 6338T	
Intel®	Xeon®	Gold 6342	
Intel®	Xeon®	Gold 6346	
Intel®	Xeon®	Gold 6348	
Intel®	Xeon®	Gold 6354	
Intel®	Xeon®	Gold Gold 5218	
Intel®	Xeon®	Gold Gold 5318Y	
Intel®	Xeon®	Gold Gold 6238R	
Intel®	Xeon®	Gold6246	
Intel®	Xeon®	Goldv 6234	
Intel®	Xeon®	Platinum 8153	
Intel®	Xeon®	Platinum 8156	
Intel®	Xeon®	Platinum 8158	
Intel®	Xeon®	Platinum 8160	
Intel®	Xeon®	Platinum 8160F	
Intel®	Xeon®	Platinum 8160T	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Platinum 8164	
Intel®	Xeon®	Platinum 8168	
Intel®	Xeon®	Platinum 8170	
Intel®	Xeon®	Platinum 8176	
Intel®	Xeon®	Platinum 8176F	
Intel®	Xeon®	Platinum 8180	
Intel®	Xeon®	Platinum 8253	
Intel®	Xeon®	Platinum 8256	
Intel®	Xeon®	Platinum 8260	
Intel®	Xeon®	Platinum 8260L	
Intel®	Xeon®	Platinum 8260Y	
Intel®	Xeon®	Platinum 8268	
Intel®	Xeon®	Platinum 8270	
Intel®	Xeon®	Platinum 8276	
Intel®	Xeon®	Platinum 8276L	
Intel®	Xeon®	Platinum 8280	
Intel®	Xeon®	Platinum 8280L	
Intel®	Xeon®	Platinum 8351N	
Intel®	Xeon®	Platinum 8352S	
Intel®	Xeon®	Platinum 8352V	
Intel®	Xeon®	Platinum 8352Y	
Intel®	Xeon®	Platinum 8358	
Intel®	Xeon®	Platinum 8358P	
Intel®	Xeon®	Platinum 8360Y	
Intel®	Xeon®	Platinum 8368	
Intel®	Xeon®	Platinum 8368Q	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Platinum 8380	
Intel®	Xeon®	Platinum 9221	
Intel®	Xeon®	Platinum 9222	
Intel®	Xeon®	Platinum 9242	
Intel®	Xeon®	Platinum 9282	
Intel®	Xeon®	Silver 4108	
Intel®	Xeon®	Silver 4109T	
Intel®	Xeon®	Silver 4110	
Intel®	Xeon®	Silver 4112	
Intel®	Xeon®	Silver 4114	
Intel®	Xeon®	Silver 4114T	
Intel®	Xeon®	Silver 4116	
Intel®	Xeon®	Silver 4116T	
Intel®	Xeon®	Silver 4208	
Intel®	Xeon®	Silver 4209T	
Intel®	Xeon®	Silver 4210	
Intel®	Xeon®	Silver 4210R	
Intel®	Xeon®	Silver 4210T	
Intel®	Xeon®	Silver 4214	
Intel®	Xeon®	Silver 4214R	
Intel®	Xeon®	Silver 4214Y	
Intel®	Xeon®	Silver 4215	
Intel®	Xeon®	Silver 4215R	
Intel®	Xeon®	Silver 4216	
Intel®	Xeon®	Silver 4309Y	
Intel®	Xeon®	Silver 4310	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	Silver 4310T	
Intel®	Xeon®	Silver 4314	
Intel®	Xeon®	Silver 4316	
Intel®	Xeon®	W-10855M	
Intel®	Xeon®	W-10885M	
Intel®	Xeon®	W-1250	
Intel®	Xeon®	W-1250E	
Intel®	Xeon®	W-1250P	
Intel®	Xeon®	W-1250TE	
Intel®	Xeon®	W-1270	
Intel®	Xeon®	W-1270E	
Intel®	Xeon®	W-1270P	
Intel®	Xeon®	W-1270TE	
Intel®	Xeon®	W-1290	
Intel®	Xeon®	W-1290E	
Intel®	Xeon®	W-1290P	
Intel®	Xeon®	W-1290T	
Intel®	Xeon®	W-1290TE	
Intel®	Xeon®	W-2123	
Intel®	Xeon®	W-2125	
Intel®	Xeon®	W-2133	
Intel®	Xeon®	W-2135	
Intel®	Xeon®	W-2145	
Intel®	Xeon®	W-2155	
Intel®	Xeon®	W-2175	
Intel®	Xeon®	W-2195	

Manufacturer	Brand	Model	IoT Applicablility
Intel®	Xeon®	W-2223	
Intel®	Xeon®	W-2225	
Intel®	Xeon®	W-2235	
Intel®	Xeon®	W-2245	
Intel®	Xeon®	W-2255	
Intel®	Xeon®	W-2265	
Intel®	Xeon®	W-2275	
Intel®	Xeon®	W-2295	
Intel®	Xeon®	W-3175X	
Intel®	Xeon®	W-3223	
Intel®	Xeon®	W-3225	
Intel®	Xeon®	W-3235	
Intel®	Xeon®	W-3245	
Intel®	Xeon®	W-3245M	
Intel®	Xeon®	W-3265	
Intel®	Xeon®	W-3265M	
Intel®	Xeon®	W-3275	
Intel®	Xeon®	W-3275M	

Windows 10 Enterprise LTSC 2021 Supported AMD Processors

Article • 05/23/2023

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model
AMD	AMD	3015e
AMD	AMD	3020e
AMD	A-Series	A10 Micro-6700T with Radeon™ R6 Graphics
AMD	A-Series	A10-6700 with Radeon™ HD 8670D
AMD	A-Series	A10-6700T with Radeon™ HD 8650D
AMD	A-Series	A10-6790B with Radeon™ HD 8670D
AMD	A-Series	A10-6790K with Radeon™ HD 8670D
AMD	A-Series	A10-6800B with Radeon™ HD 8670D
AMD	A-Series	A10-6800K with Radeon™ HD 8670D
AMD	A-Series	A10-7300 with Radeon™ R6 Graphics
AMD	A-Series	A10-7400P with Radeon™ R6 Graphics
AMD	A-Series	A10-7700K with Radeon™ R7 Series
AMD	A-Series	A10-7800 with Radeon™ R7 Series
AMD	A-Series	A10-7850K with Radeon™ R7 Series
AMD	A-Series	A10-7860K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A10-7860K with Radeon™ R7 Series
AMD	A-Series	A10-7870K with Radeon™ R7 Graphics and Near Silent Thermal Solution

Manufacturer	Brand	Model
AMD	A-Series	A10-7870K with Radeon™ R7 Series
AMD	A-Series	A10-7890K with Radeon™ R7 Graphics and Wraith cooler
AMD	A-Series	A10-8700P APU
AMD	A-Series	A10-8700P with Radeon™ R6 Graphics
AMD	A-Series	A10-9600P APU
AMD	A-Series	A10-9630P APU
AMD	A-Series	A10-9700 APU
AMD	A-Series	A10-9700E APU
AMD	A-Series	A12-9700P APU
AMD	A-Series	A12-9730P APU
AMD	A-Series	A12-9800 APU
AMD	A-Series	A12-9800E APU
AMD	A-Series	A4 Micro-6400T with Radeon™ R3 Graphics
AMD	A-Series	A4-5000 with Radeon™ HD 8330
AMD	A-Series	A4-5100 with Radeon™ HD 8330
AMD	A-Series	A4-6210 with Radeon™ R3 Graphics
AMD	A-Series	A4-6300 with Radeon™ HD 8370D
AMD	A-Series	A4-6320 with Radeon™ HD 8370D
AMD	A-Series	A4-7210 with Radeon™ R3 Graphics
AMD	A-Series	A4-7300 with Radeon™ HD 8470D
AMD	A-Series	A4-9120 APU
AMD	A-Series	A4-9120C APU
AMD	A-Series	A4-9125 APU
AMD	A-Series	A6-5200 with Radeon™ HD 8400
AMD	A-Series	A6-5200M with Radeon™ HD 8400
AMD	A-Series	A6-5350M with Radeon™ HD 8450G

Manufacturer	Brand	Model
AMD	A-Series	A6-6310 with Radeon™ R4 Graphics
AMD	A-Series	A6-6400B with Radeon™ HD 8470D
AMD	A-Series	A6-6400K with Radeon™ HD 8470D
AMD	A-Series	A6-6420B with Radeon™ HD 8470D
AMD	A-Series	A6-6420K with Radeon™ HD 8470D
AMD	A-Series	A6-7000 with Radeon™ R4 Graphics
AMD	A-Series	A6-7310 with Radeon™ R4 Graphics
AMD	A-Series	A6-7400K with Radeon™ R5 Series
AMD	A-Series	A6-7470K with Radeon™ R5 Series
AMD	A-Series	A6-8500P with Radeon™ R5 Graphics
AMD	A-Series	A6-9210 APU
AMD	A-Series	A6-9220 APU
AMD	A-Series	A6-9220C APU
AMD	A-Series	A6-9225 APU
AMD	A-Series	A6-9500 APU
AMD	A-Series	A6-9500E APU
AMD	A-Series	A6-9550 APU
AMD	A-Series	A8-6410 with Radeon™ R5 Graphics
AMD	A-Series	A8-6500 with Radeon™ HD 8570D
AMD	A-Series	A8-6500B with Radeon™ HD 8570D
AMD	A-Series	A8-6500T with Radeon™ HD 8550D
AMD	A-Series	A8-6600K with Radeon™ HD 8570D
AMD	A-Series	A8-7100 with Radeon™ R5 Graphics
AMD	A-Series	A8-7200P with Radeon™ R5 Graphics
AMD	A-Series	A8-7410 with Radeon™ R5 Graphics
AMD	A-Series	A8-7600 with Radeon™ R7 Series

Manufacturer	Brand	Model
AMD	A-Series	A8-7650K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-7650K with Radeon™ R7 Series
AMD	A-Series	A8-7670K with Radeon™ R7 Graphics and Near Silent Thermal Solution
AMD	A-Series	A8-8600P APU
AMD	A-Series	A8-8600P with Radeon™ R6 Graphics
AMD	A-Series	A8-9600 APU
AMD	A-Series	A9-9410 APU
AMD	A-Series	A9-9420 APU
AMD	A-Series	A9-9425 APU
AMD	Athlon™	200GE
AMD	Athlon™	220GE
AMD	Athlon™	240GE
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	Athlon™ 5150 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5350 APU with Radeon™ R3 Series
AMD	Athlon™	Athlon™ 5370 APU with Radeon™ R3 Series
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G
AMD	Athlon™	Gold 3150GE
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e

Manufacturer	Brand	Model
AMD	Athlon™	Silver 3050GE
AMD	Athlon™	Silver 3050U
AMD	Athlon™	X4 940
AMD	Athlon™	X4 950
AMD	Athlon™	X4 970
AMD	EPYC™	7251
AMD	EPYC™	7252
AMD	EPYC™	7261
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7281
AMD	EPYC™	7282
AMD	EPYC™	7301
AMD	EPYC™	7302
AMD	EPYC™	7313
AMD	EPYC™	7343
AMD	EPYC™	7351
AMD	EPYC™	7352
AMD	EPYC™	7371
AMD	EPYC™	7401
AMD	EPYC™	7402
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7451
AMD	EPYC™	7452
AMD	EPYC™	7453

Manufacturer	Brand	Model
AMD	EPYC™	7501
AMD	EPYC™	7502
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7551
AMD	EPYC™	7552
AMD	EPYC™	7601
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7713
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7232P
AMD	EPYC™	72F3
AMD	EPYC™	7302P
AMD	EPYC™	7313P
AMD	EPYC™	7351P
AMD	EPYC™	73F3
AMD	EPYC™	7401P
AMD	EPYC™	7402P
AMD	EPYC™	7443P

Manufacturer	Brand	Model
AMD	EPYC™	74F3
AMD	EPYC™	7502P
AMD	EPYC™	7543P
AMD	EPYC™	7551P
AMD	EPYC™	75F3
AMD	EPYC™	7702P
AMD	EPYC™	7713P
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	EPYC™	8024P
AMD	EPYC™	8024PN
AMD	EPYC™	8124P
AMD	EPYC™	8124PN
AMD	EPYC™	8224P
AMD	EPYC™	8224PN
AMD	EPYC™	8324P
AMD	EPYC™	8324PN
AMD	EPYC™	8434P
AMD	EPYC™	8434PN
AMD	EPYC™	8534P
AMD	EPYC™	8534PN
AMD	E-Series	E1 Micro-6200T with Radeon™ R2 Graphics
AMD	E-Series	E1-2100 with Radeon™ HD 8210
AMD	E-Series	E1-2200 with Radeon™ HD 8210

Manufacturer	Brand	Model
AMD	E-Series	E1-2500 with Radeon™ HD 8240
AMD	E-Series	E1-6010 with Radeon™ R2 Graphics
AMD	E-Series	E1-7010 with Radeon™ R2 Graphics
AMD	E-Series	E2-3000 with Radeon™ HD 8280
AMD	E-Series	E2-3800 with Radeon™ HD 8280
AMD	E-Series	E2-6110 with Radeon™ R2 Graphics
AMD	E-Series	E2-7110 with Radeon™ R2 Graphics
AMD	E-Series	E2-9010 APU
AMD	FX-Series	9800P APU
AMD	FX-Series	9830P APU
AMD	FX-Series	FX 6100
AMD	FX-Series	FX-4100
AMD	FX-Series	FX-4130
AMD	FX-Series	FX-4170
AMD	FX-Series	FX-4300
AMD	FX-Series	FX-4320
AMD	FX-Series	FX-4350
AMD	FX-Series	FX-6200
AMD	FX-Series	FX-6300
AMD	FX-Series	FX-6350
AMD	FX-Series	FX-6350 with Wraith cooler
AMD	FX-Series	FX-7500 with Radeon™ R7 Graphics
AMD	FX-Series	FX-7600P with Radeon™ R7 Graphics
AMD	FX-Series	FX-8120
AMD	FX-Series	FX-8150
AMD	FX-Series	FX-8300

Manufacturer	Brand	Model
AMD	FX-Series	FX-8310
AMD	FX-Series	FX-8320
AMD	FX-Series	FX-8320E
AMD	FX-Series	FX-8350
AMD	FX-Series	FX-8350 with Wraith cooler
AMD	FX-Series	FX-8370
AMD	FX-Series	FX-8370 with Wraith cooler
AMD	FX-Series	FX-8370E
AMD	FX-Series	FX-8800P APU
AMD	FX-Series	FX-8800P with Radeon™ R7 Graphics
AMD	FX-Series	FX-9370
AMD	FX-Series	FX-9590
AMD	PRO A-Series	A10 PRO-7350B with Radeon™ R6 Graphics
AMD	PRO A-Series	A10 PRO-7800B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10 PRO-7850B with Radeon™ R7 Graphics
AMD	PRO A-Series	A10-8700B APU
AMD	PRO A-Series	A10-8730B APU
AMD	PRO A-Series	A10-8750B APU
AMD	PRO A-Series	A10-8770 APU
AMD	PRO A-Series	A10-8770E APU
AMD	PRO A-Series	A10-8850B APU
AMD	PRO A-Series	A10-9700 APU
AMD	PRO A-Series	A10-9700B APU
AMD	PRO A-Series	A10-9700E APU
AMD	PRO A-Series	A10-9730B APU
AMD	PRO A-Series	A12-8800B APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A12-8830B APU
AMD	PRO A-Series	A12-8870 APU
AMD	PRO A-Series	A12-8870E APU
AMD	PRO A-Series	A12-9800 APU
AMD	PRO A-Series	A12-9800B APU
AMD	PRO A-Series	A12-9800E APU
AMD	PRO A-Series	A12-9830B APU
AMD	PRO A-Series	A4 PRO-3340B with Radeon™ HD 8240 Graphics
AMD	PRO A-Series	A4 PRO-3350B with Radeon™ R4 Graphics
AMD	PRO A-Series	A4 PRO-7300B with Radeon™ HD 8470D
AMD	PRO A-Series	A4 PRO-7350B with Radeon™ R5 Graphics
AMD	PRO A-Series	A4-4350B APU
AMD	PRO A-Series	A4-5350B APU
AMD	PRO A-Series	A4-8350B APU
AMD	PRO A-Series	A6 PRO-7050B with Radeon™ R4 Graphics
AMD	PRO A-Series	A6 PRO-7400B with Radeon™ R5 Graphics
AMD	PRO A-Series	A6-7350B APU
AMD	PRO A-Series	A6-8350B APU
AMD	PRO A-Series	A6-8500B APU
AMD	PRO A-Series	A6-8530B APU
AMD	PRO A-Series	A6-8550B APU
AMD	PRO A-Series	A6-8570 APU
AMD	PRO A-Series	A6-8570E APU
AMD	PRO A-Series	A6-9500 APU
AMD	PRO A-Series	A6-9500B APU
AMD	PRO A-Series	A6-9500E APU

Manufacturer	Brand	Model
AMD	PRO A-Series	A8 PRO-7150B with Radeon™ R5 Graphics
AMD	PRO A-Series	A8 PRO-7600B with Radeon™ R7 Graphics
AMD	PRO A-Series	A8-8600B APU
AMD	PRO A-Series	A8-8650B APU
AMD	PRO A-Series	A8-9600 APU
AMD	PRO A-Series	A8-9600B APU
AMD	PRO A-Series	A8-9630B
AMD	Ryzen™ 3	1200
AMD	Ryzen™ 3	3100
AMD	Ryzen™ 3	1300X
AMD	Ryzen™ 3	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	2200U
AMD	Ryzen™ 3	2300U
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G
AMD	Ryzen™ 3	4300GE
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3	5300U

Manufacturer	Brand	Model
AMD	Ryzen™ 3	5400U
AMD	Ryzen™ 3 PRO	1200
AMD	Ryzen™ 3 PRO	1300
AMD	Ryzen™ 3 PRO	2200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2200GE with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3 PRO	2300U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 3 PRO	5350G
AMD	Ryzen™ 3 PRO	5350GE
AMD	Ryzen™ 3 PRO	5450U
AMD	Ryzen™ 5	1400
AMD	Ryzen™ 5	1600
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	1500X
AMD	Ryzen™ 5	1600 (AF)
AMD	Ryzen™ 5	1600X
AMD	Ryzen™ 5	2400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2400GE with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	2500U
AMD	Ryzen™ 5	2500X

Manufacturer	Brand	Model
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600H
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G
AMD	Ryzen™ 5	4600GE
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5	5300G
AMD	Ryzen™ 5	5300GE
AMD	Ryzen™ 5	5500U
AMD	Ryzen™ 5	5600G
AMD	Ryzen™ 5	5600GE
AMD	Ryzen™ 5	5600H
AMD	Ryzen™ 5	5600HS
AMD	Ryzen™ 5	5600U

Manufacturer	Brand	Model
AMD	Ryzen™ 5	5600X
AMD	Ryzen™ 5 PRO	1500
AMD	Ryzen™ 5 PRO	1600
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	2400G with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2400GE with Radeon™ Vega 11 Graphics
AMD	Ryzen™ 5 PRO	2500U
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 5 PRO	5650G
AMD	Ryzen™ 5 PRO	5650GE
AMD	Ryzen™ 5 PRO	5650U
AMD	Ryzen™ 5 PRO	5750G
AMD	Ryzen™ 5 PRO	5750GE
AMD	Ryzen™ 7	1700
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	1700X
AMD	Ryzen™ 7	1800X
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700U

Manufacturer	Brand	Model
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	2800H
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G
AMD	Ryzen™ 7	4700GE
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7	5700G
AMD	Ryzen™ 7	5700GE
AMD	Ryzen™ 7	5700U
AMD	Ryzen™ 7	5800H
AMD	Ryzen™ 7	5800HS
AMD	Ryzen™ 7	5800U
AMD	Ryzen™ 7	5800X
AMD	Ryzen™ 7 PRO	1700
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	1700X
AMD	Ryzen™ 7 PRO	2700U

Manufacturer	Brand	Model
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 7 PRO	5850U
AMD	Ryzen™ 9	5900
AMD	Ryzen™ 9	3900 Processor
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9	5900HS
AMD	Ryzen™ 9	5900HX
AMD	Ryzen™ 9	5900X
AMD	Ryzen™ 9	5950X
AMD	Ryzen™ 9	5980HS
AMD	Ryzen™ 9	5980HX
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ Threadripper™	1900X
AMD	Ryzen™ Threadripper™	1920X
AMD	Ryzen™ Threadripper™	1950X
AMD	Ryzen™ Threadripper™	2920X
AMD	Ryzen™ Threadripper™	2950X
AMD	Ryzen™ Threadripper™	2970WX

Manufacturer	Brand	Model
AMD	Ryzen™ Threadripper™	2990WX
AMD	Ryzen™ Threadripper™	3960X
AMD	Ryzen™ Threadripper™	3970X
AMD	Ryzen™ Threadripper™	3990X
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Windows 10 Enterprise LTSC 2021 Supported Intel Processors

Article • 09/09/2024

Earlier generations and models of the CPUs listed may have limited support for devices on this version of Windows 10. Please contact your hardware vendor(s) for specific support details.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Intel®	Atom®	E3805	IoT Enterprise Only
Intel®	Atom®	E3827	IoT Enterprise Only
Intel®	Atom®	E3826	IoT Enterprise Only
Intel®	Atom®	E3815	IoT Enterprise Only
Intel®	Atom®	E3845	IoT Enterprise Only
Intel®	Atom®	E3825	IoT Enterprise Only
Intel®	Atom®	x5-E3930	IoT Enterprise Only
Intel®	Atom®	x5-E3940	IoT Enterprise Only
Intel®	Atom®	x7-E3950	IoT Enterprise Only
Intel®	Atom®	x5-E8000	IoT Enterprise Only
Intel®	Atom®	x6200FE	
Intel®	Atom®	x6211E	
Intel®	Atom®	x6212RE	
Intel®	Atom®	x6413E	
Intel®	Atom®	x6414RE	
Intel®	Atom®	x6425E	
Intel®	Atom®	x6425RE	
Intel®	Atom®	x6427FE	
Intel®	Atom®	x7203C	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Atom®	x7211E	IoT Enterprise Only
Intel®	Atom®	x7211RE	IoT Enterprise Only
Intel®	Atom®	x7213E	IoT Enterprise Only
Intel®	Atom®	x7213RE	IoT Enterprise Only
Intel®	Atom®	x7405C	IoT Enterprise Only
Intel®	Atom®	x7425E	IoT Enterprise Only
Intel®	Atom®	x7433RE	IoT Enterprise Only
Intel®	Atom®	x7809C	IoT Enterprise Only
Intel®	Atom®	x7835RE	IoT Enterprise Only
Intel®	Celeron®	3205U	
Intel®	Celeron®	3215U	
Intel®	Celeron®	3755U	
Intel®	Celeron®	3765U	
Intel®	Celeron®	3855U	
Intel®	Celeron®	3865U	
Intel®	Celeron®	3867U	
Intel®	Celeron®	3955U	
Intel®	Celeron®	3965U	
Intel®	Celeron®	3965Y	
Intel®	Celeron®	4205U	
Intel®	Celeron®	4305U	
Intel®	Celeron®	4305UE	
Intel®	Celeron®	5205U	
Intel®	Celeron®	5305U	
Intel®	Celeron®	6305	
Intel®	Celeron®	6305E	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	7305E	IoT Enterprise Only
Intel®	Celeron®	7305L	IoT Enterprise Only
Intel®	Celeron®	G3900	
Intel®	Celeron®	G3900E	
Intel®	Celeron®	G3900T	
Intel®	Celeron®	G3900TE	
Intel®	Celeron®	G3902E	
Intel®	Celeron®	G3920	
Intel®	Celeron®	G3930	
Intel®	Celeron®	G3930E	
Intel®	Celeron®	G3930T	
Intel®	Celeron®	G3930TE	
Intel®	Celeron®	G3950	
Intel®	Celeron®	G4900	
Intel®	Celeron®	G4900T	
Intel®	Celeron®	G4920	
Intel®	Celeron®	G4930	
Intel®	Celeron®	G4930E	
Intel®	Celeron®	G4930T	
Intel®	Celeron®	G4932E	
Intel®	Celeron®	G4950	
Intel®	Celeron®	G5900	
Intel®	Celeron®	G5900E	
Intel®	Celeron®	G5900T	
Intel®	Celeron®	G5900TE	
Intel®	Celeron®	G5905	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	G5905T	
Intel®	Celeron®	G5920	
Intel®	Celeron®	G5925	
Intel®	Celeron®	G6900E	IoT Enterprise Only
Intel®	Celeron®	G6900TE	IoT Enterprise Only
Intel®	Celeron®	J1900	IoT Enterprise Only
Intel®	Celeron®	J3355	IoT Enterprise Only
Intel®	Celeron®	J3355E	IoT Enterprise Only
Intel®	Celeron®	J3455	IoT Enterprise Only
Intel®	Celeron®	J3455E	IoT Enterprise Only
Intel®	Celeron®	J4005	
Intel®	Celeron®	J4025	
Intel®	Celeron®	J4105	
Intel®	Celeron®	J4115	
Intel®	Celeron®	J4125	
Intel®	Celeron®	J6412	
Intel®	Celeron®	J6413	
Intel®	Celeron®	N2807	IoT Enterprise Only
Intel®	Celeron®	N2920	IoT Enterprise Only
Intel®	Celeron®	N2930	IoT Enterprise Only
Intel®	Celeron®	N3010	IoT Enterprise Only
Intel®	Celeron®	N3060	IoT Enterprise Only
Intel®	Celeron®	N3160	IoT Enterprise Only
Intel®	Celeron®	N3350	IoT Enterprise Only
Intel®	Celeron®	N4000	
Intel®	Celeron®	N4020	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Celeron®	N4100	
Intel®	Celeron®	N4120	
Intel®	Celeron®	N4500	
Intel®	Celeron®	N4505	
Intel®	Celeron®	N5100	
Intel®	Celeron®	N5105	
Intel®	Celeron®	N6210	
Intel®	Celeron®	N6211	
Intel®	Core™	3-100HL	IoT Enterprise Only
Intel®	Core™	3-100U	IoT Enterprise Only
Intel®	Core™	3-100UL	IoT Enterprise Only
Intel®	Core™	5-120HL	IoT Enterprise Only
Intel®	Core™	5-120U	IoT Enterprise Only
Intel®	Core™	5-120UL	IoT Enterprise Only
Intel®	Core™	5-130HL	IoT Enterprise Only
Intel®	Core™	5-130UL	IoT Enterprise Only
Intel®	Core™	7-150HL	IoT Enterprise Only
Intel®	Core™	7-150U	IoT Enterprise Only
Intel®	Core™	7-150UL	IoT Enterprise Only
Intel®	Core™	7-160HL	IoT Enterprise Only
Intel®	Core™	7-160UL	IoT Enterprise Only
Intel®	Core™	i3-1000G1	
Intel®	Core™	i3-1000G4	
Intel®	Core™	i3-1005G1	
Intel®	Core™	i3-10100	
Intel®	Core™	i3-10100E	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-10100F	
Intel®	Core™	i3-10100T	
Intel®	Core™	i3-10100TE	
Intel®	Core™	i3-10100Y	
Intel®	Core™	i3-10105	
Intel®	Core™	i3-10105F	
Intel®	Core™	i3-10105T	
Intel®	Core™	i3-10110U	
Intel®	Core™	i3-10110Y	
Intel®	Core™	i3-10300	
Intel®	Core™	i3-10300T	
Intel®	Core™	i3-10305	
Intel®	Core™	i3-10305T	
Intel®	Core™	i3-10320	
Intel®	Core™	i3-10325	
Intel®	Core™	i3-1110G4	
Intel®	Core™	i3-1115G4	
Intel®	Core™	i3-1115G4E	
Intel®	Core™	i3-1115GRE	
Intel®	Core™	i3-1120G4	
Intel®	Core™	i3-1125G4	
Intel®	Core™	i3-12100	
Intel®	Core™	i3-12100E	IoT Enterprise Only
Intel®	Core™	i3-12100TE	IoT Enterprise Only
Intel®	Core™	i3-1215UE	IoT Enterprise Only
Intel®	Core™	i3-1220PE	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-12300HE	IoT Enterprise Only
Intel®	Core™	i3-1305U	IoT Enterprise Only
Intel®	Core™	i3-13100	IoT Enterprise Only
Intel®	Core™	i3-13100E	IoT Enterprise Only
Intel®	Core™	i3-13100F	IoT Enterprise Only
Intel®	Core™	i3-13100T	IoT Enterprise Only
Intel®	Core™	i3-13100TE	IoT Enterprise Only
Intel®	Core™	i3-1315U	IoT Enterprise Only
Intel®	Core™	i3-1315UE	IoT Enterprise Only
Intel®	Core™	i3-1315URE	IoT Enterprise Only
Intel®	Core™	i3-1320PE	IoT Enterprise Only
Intel®	Core™	i3-1320PRE	IoT Enterprise Only
Intel®	Core™	i3-13300HE	IoT Enterprise Only
Intel®	Core™	i3-13300HRE	IoT Enterprise Only
Intel®	Core™	i3-14100	IoT Enterprise Only
Intel®	Core™	i3-14100F	IoT Enterprise Only
Intel®	Core™	i3-14100T	IoT Enterprise Only
Intel®	Core™	i3-5005U	
Intel®	Core™	i3-5010U	
Intel®	Core™	i3-5015U	
Intel®	Core™	i3-5020U	
Intel®	Core™	i3-5157U	
Intel®	Core™	i3-6006U	
Intel®	Core™	i3-6098P	
Intel®	Core™	i3-6100	
Intel®	Core™	i3-6100E	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-6100H	
Intel®	Core™	i3-6100T	
Intel®	Core™	i3-6100TE	
Intel®	Core™	i3-6100U	
Intel®	Core™	i3-6102E	
Intel®	Core™	i3-6157U	
Intel®	Core™	i3-6167U	
Intel®	Core™	i3-6300	
Intel®	Core™	i3-6300T	
Intel®	Core™	i3-6320	
Intel®	Core™	i3-7020U	
Intel®	Core™	i3-7100	
Intel®	Core™	i3-7100E	
Intel®	Core™	i3-7100H	
Intel®	Core™	i3-7100T	
Intel®	Core™	i3-7100U	
Intel®	Core™	i3-7101E	
Intel®	Core™	i3-7101TE	
Intel®	Core™	i3-7102E	
Intel®	Core™	i3-7130U	
Intel®	Core™	i3-7167U	
Intel®	Core™	i3-7300	
Intel®	Core™	i3-7300T	
Intel®	Core™	i3-7320	
Intel®	Core™	i3-7350K	
Intel®	Core™	i3-8100	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i3-8100B	
Intel®	Core™	i3-8100H	
Intel®	Core™	i3-8100T	
Intel®	Core™	i3-8109U	
Intel®	Core™	i3-8130U	
Intel®	Core™	i3-8140U	
Intel®	Core™	i3-8145U	
Intel®	Core™	i3-8145UE	
Intel®	Core™	i3-8300	
Intel®	Core™	i3-8300T	
Intel®	Core™	i3-8350K	
Intel®	Core™	i3-9100	
Intel®	Core™	i3-9100E	
Intel®	Core™	i3-9100F	
Intel®	Core™	i3-9100HL	
Intel®	Core™	i3-9100T	
Intel®	Core™	i3-9100TE	
Intel®	Core™	i3-9300	
Intel®	Core™	i3-9300T	
Intel®	Core™	i3-9320	
Intel®	Core™	i3-9350K	
Intel®	Core™	i3-9350KF	
Intel®	Core™	i3-L13G4	
Intel®	Core™	i3-N300	IoT Enterprise Only
Intel®	Core™	i3-N305	IoT Enterprise Only
Intel®	Core™	i5-10200H	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-10210U	
Intel®	Core™	i5-10210Y	
Intel®	Core™	i5-10300H	
Intel®	Core™	i5-1030G4	
Intel®	Core™	i5-1030G7	
Intel®	Core™	i5-10310U	
Intel®	Core™	i5-10310Y	
Intel®	Core™	i5-1035G1	
Intel®	Core™	i5-1035G4	
Intel®	Core™	i5-1035G7	
Intel®	Core™	i5-1038NG7	
Intel®	Core™	i5-10400	
Intel®	Core™	i5-10400F	
Intel®	Core™	i5-10400H	
Intel®	Core™	i5-10400T	
Intel®	Core™	i5-10500	
Intel®	Core™	i5-10500E	
Intel®	Core™	i5-10500H	
Intel®	Core™	i5-10500T	
Intel®	Core™	i5-10500TE	
Intel®	Core™	i5-10505	
Intel®	Core™	i5-10600	
Intel®	Core™	i5-10600K	
Intel®	Core™	i5-10600KF	
Intel®	Core™	i5-10600T	
Intel®	Core™	i5-11260H	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-11300H	
Intel®	Core™	i5-1130G7	
Intel®	Core™	i5-11320H	
Intel®	Core™	i5-1135G7	
Intel®	Core™	i5-11400	
Intel®	Core™	i5-11400F	
Intel®	Core™	i5-11400H	
Intel®	Core™	i5-11400T	
Intel®	Core™	i5-1140G7	
Intel®	Core™	i5-1145G7	
Intel®	Core™	i5-1145G7E	
Intel®	Core™	i5-1145GRE	
Intel®	Core™	i5-11500	
Intel®	Core™	i5-11500H	
Intel®	Core™	i5-11500T	
Intel®	Core™	i5-1155G7	
Intel®	Core™	i5-11600	
Intel®	Core™	i5-11600K	
Intel®	Core™	i5-11600KF	
Intel®	Core™	i5-11600T	
Intel®	Core™	i5-1235U	
Intel®	Core™	i5-12400	
Intel®	Core™	i5-1245UE	IoT Enterprise Only
Intel®	Core™	i5-12500	
Intel®	Core™	i5-12500E	IoT Enterprise Only
Intel®	Core™	i5-12500TE	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-1250PE	IoT Enterprise Only
Intel®	Core™	i5-12600HE	IoT Enterprise Only
Intel®	Core™	i5-12600K	
Intel®	Core™	i5-12600KF	
Intel®	Core™	i5-1334U	IoT Enterprise Only
Intel®	Core™	i5-1335U	IoT Enterprise Only
Intel®	Core™	i5-1335UE	IoT Enterprise Only
Intel®	Core™	i5-13400	
Intel®	Core™	i5-13400E	IoT Enterprise Only
Intel®	Core™	i5-13400F	IoT Enterprise Only
Intel®	Core™	i5-13400T	
Intel®	Core™	i5-1340P	IoT Enterprise Only
Intel®	Core™	i5-1340PE	IoT Enterprise Only
Intel®	Core™	i5-13420H	IoT Enterprise Only
Intel®	Core™	i5-13450HX	IoT Enterprise Only
Intel®	Core™	i5-1345U	IoT Enterprise Only
Intel®	Core™	i5-1345UE	IoT Enterprise Only
Intel®	Core™	i5-1345URE	IoT Enterprise Only
Intel®	Core™	i5-13500	
Intel®	Core™	i5-13500E	IoT Enterprise Only
Intel®	Core™	i5-13500H	IoT Enterprise Only
Intel®	Core™	i5-13500HX	IoT Enterprise Only
Intel®	Core™	i5-13500T	
Intel®	Core™	i5-13500TE	IoT Enterprise Only
Intel®	Core™	i5-13505H	IoT Enterprise Only
Intel®	Core™	i5-1350P	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-1350PE	IoT Enterprise Only
Intel®	Core™	i5-1350PRE	IoT Enterprise Only
Intel®	Core™	i5-13600	
Intel®	Core™	i5-13600H	IoT Enterprise Only
Intel®	Core™	i5-13600HE	IoT Enterprise Only
Intel®	Core™	i5-13600HRE	IoT Enterprise Only
Intel®	Core™	i5-13600HX	IoT Enterprise Only
Intel®	Core™	i5-13600K	
Intel®	Core™	i5-13600KF	IoT Enterprise Only
Intel®	Core™	i5-13600T	
Intel®	Core™	i5-14400	IoT Enterprise Only
Intel®	Core™	i5-14400F	IoT Enterprise Only
Intel®	Core™	i5-14400T	IoT Enterprise Only
Intel®	Core™	i5-14450HX	IoT Enterprise Only
Intel®	Core™	i5-14500	IoT Enterprise Only
Intel®	Core™	i5-14500HX	IoT Enterprise Only
Intel®	Core™	i5-14500T	IoT Enterprise Only
Intel®	Core™	i5-14600	IoT Enterprise Only
Intel®	Core™	i5-14600K	IoT Enterprise Only
Intel®	Core™	i5-14600KF	IoT Enterprise Only
Intel®	Core™	i5-14600T	IoT Enterprise Only
Intel®	Core™	i5-5200U	
Intel®	Core™	i5-5250U	
Intel®	Core™	i5-5257U	
Intel®	Core™	i5-5287U	
Intel®	Core™	i5-5300U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-5350H	
Intel®	Core™	i5-5350U	
Intel®	Core™	i5-5575R	
Intel®	Core™	i5-5675C	
Intel®	Core™	i5-5675R	
Intel®	Core™	i5-6200U	
Intel®	Core™	i5-6260U	
Intel®	Core™	i5-6267U	
Intel®	Core™	i5-6287U	
Intel®	Core™	i5-6300HQ	
Intel®	Core™	i5-6300U	
Intel®	Core™	i5-6350HQ	
Intel®	Core™	i5-6360U	
Intel®	Core™	i5-6400	
Intel®	Core™	i5-6400T	
Intel®	Core™	i5-6402P	
Intel®	Core™	i5-6440EQ	
Intel®	Core™	i5-6440HQ	
Intel®	Core™	i5-6442EQ	
Intel®	Core™	i5-6500	
Intel®	Core™	i5-6500T	
Intel®	Core™	i5-6500TE	
Intel®	Core™	i5-6585R	
Intel®	Core™	i5-6600	
Intel®	Core™	i5-6600K	
Intel®	Core™	i5-6600T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-6685R	
Intel®	Core™	i5-7200U	
Intel®	Core™	i5-7260U	
Intel®	Core™	i5-7267U	
Intel®	Core™	i5-7287U	
Intel®	Core™	i5-7300HQ	
Intel®	Core™	i5-7300U	
Intel®	Core™	i5-7360U	
Intel®	Core™	i5-7400	
Intel®	Core™	i5-7400T	
Intel®	Core™	i5-7440EQ	
Intel®	Core™	i5-7440HQ	
Intel®	Core™	i5-7442EQ	
Intel®	Core™	i5-7500	
Intel®	Core™	i5-7500T	
Intel®	Core™	i5-7600	
Intel®	Core™	i5-7600K	
Intel®	Core™	i5-7600T	
Intel®	Core™	i5-7640X	
Intel®	Core™	i5-7Y54	
Intel®	Core™	i5-7Y57	
Intel®	Core™	i5-8200Y	
Intel®	Core™	i5-8210Y	
Intel®	Core™	i5-8250U	
Intel®	Core™	i5-8257U	
Intel®	Core™	i5-8259U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-8260U	
Intel®	Core™	i5-8265U	
Intel®	Core™	i5-8269U	
Intel®	Core™	i5-8279U	
Intel®	Core™	i5-8300H	
Intel®	Core™	i5-8305G	
Intel®	Core™	i5-8310Y	
Intel®	Core™	i5-8350U	
Intel®	Core™	i5-8365U	
Intel®	Core™	i5-8365UE	
Intel®	Core™	i5-8400	
Intel®	Core™	i5-8400B	
Intel®	Core™	i5-8400H	
Intel®	Core™	i5-8400T	
Intel®	Core™	i5-8500	
Intel®	Core™	i5-8500B	
Intel®	Core™	i5-8500T	
Intel®	Core™	i5-8600	
Intel®	Core™	i5-8600K	
Intel®	Core™	i5-8600T	
Intel®	Core™	i5-9300H	
Intel®	Core™	i5-9300HF	
Intel®	Core™	i5-9400	
Intel®	Core™	i5-9400F	
Intel®	Core™	i5-9400H	
Intel®	Core™	i5-9400T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i5-9500	
Intel®	Core™	i5-9500E	
Intel®	Core™	i5-9500F	
Intel®	Core™	i5-9500T	
Intel®	Core™	i5-9500TE	
Intel®	Core™	i5-9600	
Intel®	Core™	i5-9600K	
Intel®	Core™	i5-9600KF	
Intel®	Core™	i5-9600T	
Intel®	Core™	i5-L16G7	
Intel®	Core™	i7-10510U	
Intel®	Core™	i7-10510Y	
Intel®	Core™	i7-1060G7	
Intel®	Core™	i7-10610U	
Intel®	Core™	i7-1065G7	
Intel®	Core™	i7-1068NG7	
Intel®	Core™	i7-10700	
Intel®	Core™	i7-10700E	
Intel®	Core™	i7-10700F	
Intel®	Core™	i7-10700K	
Intel®	Core™	i7-10700KF	
Intel®	Core™	i7-10700T	
Intel®	Core™	i7-10700TE	
Intel®	Core™	i7-10710U	
Intel®	Core™	i7-10750H	
Intel®	Core™	i7-10810U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-10850H	
Intel®	Core™	i7-10870H	
Intel®	Core™	i7-10875H	
Intel®	Core™	i7-11370H	
Intel®	Core™	i7-11375H	
Intel®	Core™	i7-11390H	
Intel®	Core™	i7-11600H	
Intel®	Core™	i7-1160G7	
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-1165G7	
Intel®	Core™	i7-11700	
Intel®	Core™	i7-11700F	
Intel®	Core™	i7-11700K	
Intel®	Core™	i7-11700KF	
Intel®	Core™	i7-11700T	
Intel®	Core™	i7-11800H	
Intel®	Core™	i7-1180G7	
Intel®	Core™	i7-11850H	
Intel®	Core™	i7-1185G7	
Intel®	Core™	i7-1185G7E	
Intel®	Core™	i7-1185GRE	
Intel®	Core™	i7-1195G7	
Intel®	Core™	i7-1265UE	IoT Enterprise Only
Intel®	Core™	i7-12700	
Intel®	Core™	i7-12700E	IoT Enterprise Only
Intel®	Core™	i7-12700K	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-12700KF	
Intel®	Core™	i7-12700TE	IoT Enterprise Only
Intel®	Core™	i7-1270PE	IoT Enterprise Only
Intel®	Core™	i7-12800HE	IoT Enterprise Only
Intel®	Core™	i7-1355U	IoT Enterprise Only
Intel®	Core™	i7-1360P	IoT Enterprise Only
Intel®	Core™	i7-1365U	IoT Enterprise Only
Intel®	Core™	i7-1365UE	IoT Enterprise Only
Intel®	Core™	i7-1365URE	IoT Enterprise Only
Intel®	Core™	i7-1366URE	IoT Enterprise Only
Intel®	Core™	i7-13620H	IoT Enterprise Only
Intel®	Core™	i7-13650HX	IoT Enterprise Only
Intel®	Core™	i7-13700	IoT Enterprise Only
Intel®	Core™	i7-13700E	IoT Enterprise Only
Intel®	Core™	i7-13700F	IoT Enterprise Only
Intel®	Core™	i7-13700H	IoT Enterprise Only
Intel®	Core™	i7-13700HX	IoT Enterprise Only
Intel®	Core™	i7-13700K	IoT Enterprise Only
Intel®	Core™	i7-13700KF	IoT Enterprise Only
Intel®	Core™	i7-13700T	IoT Enterprise Only
Intel®	Core™	i7-13700TE	IoT Enterprise Only
Intel®	Core™	i7-13705H	IoT Enterprise Only
Intel®	Core™	i7-1370P	IoT Enterprise Only
Intel®	Core™	i7-1370PE	IoT Enterprise Only
Intel®	Core™	i7-1370PRE	IoT Enterprise Only
Intel®	Core™	i7-1375PRE	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-13800H	IoT Enterprise Only
Intel®	Core™	i7-13800HE	IoT Enterprise Only
Intel®	Core™	i7-13800HRE	IoT Enterprise Only
Intel®	Core™	i7-13850HX	IoT Enterprise Only
Intel®	Core™	i7-13700	
Intel®	Core™	i7-13700K	
Intel®	Core™	i7-13700T	
Intel®	Core™	i7-14650HX	IoT Enterprise Only
Intel®	Core™	i7-14700	IoT Enterprise Only
Intel®	Core™	i7-14700F	IoT Enterprise Only
Intel®	Core™	i7-14700HX	IoT Enterprise Only
Intel®	Core™	i7-14700K	IoT Enterprise Only
Intel®	Core™	i7-14700KF	IoT Enterprise Only
Intel®	Core™	i7-14700T	IoT Enterprise Only
Intel®	Core™	i7-5500U	
Intel®	Core™	i7-5550U	
Intel®	Core™	i7-5557U	
Intel®	Core™	i7-5600U	
Intel®	Core™	i7-5650U	
Intel®	Core™	i7-5700EQ	
Intel®	Core™	i7-5700HQ	
Intel®	Core™	i7-5750HQ	
Intel®	Core™	i7-5775C	
Intel®	Core™	i7-5775R	
Intel®	Core™	i7-5850EQ	
Intel®	Core™	i7-5850HQ	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-5950HQ	
Intel®	Core™	i7-6500U	
Intel®	Core™	i7-6560U	
Intel®	Core™	i7-6567U	
Intel®	Core™	i7-6600U	
Intel®	Core™	i7-6650U	
Intel®	Core™	i7-6660U	
Intel®	Core™	i7-6700	
Intel®	Core™	i7-6700HQ	
Intel®	Core™	i7-6700K	
Intel®	Core™	i7-6700T	
Intel®	Core™	i7-6700TE	
Intel®	Core™	i7-6770HQ	
Intel®	Core™	i7-6785R	
Intel®	Core™	i7-6820EQ	
Intel®	Core™	i7-6820HK	
Intel®	Core™	i7-6820HQ	
Intel®	Core™	i7-6822EQ	
Intel®	Core™	i7-6870HQ	
Intel®	Core™	i7-6920HQ	
Intel®	Core™	i7-6970HQ	
Intel®	Core™	i7-7500U	
Intel®	Core™	i7-7560U	
Intel®	Core™	i7-7567U	
Intel®	Core™	i7-7600U	
Intel®	Core™	i7-7660U	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-7700	
Intel®	Core™	i7-7700HQ	
Intel®	Core™	i7-7700K	
Intel®	Core™	i7-7700T	
Intel®	Core™	i7-7740X	
Intel®	Core™	i7-7800X	
Intel®	Core™	i7-7820EQ	
Intel®	Core™	i7-7820HK	
Intel®	Core™	i7-7820HQ	
Intel®	Core™	i7-7820X	
Intel®	Core™	i7-7920HQ	
Intel®	Core™	i7-7Y75	
Intel®	Core™	i7-8086K	
Intel®	Core™	i7-8500Y	
Intel®	Core™	i7-8550U	
Intel®	Core™	i7-8557U	
Intel®	Core™	i7-8559U	
Intel®	Core™	i7-8565U	
Intel®	Core™	i7-8569U	
Intel®	Core™	i7-8650U	
Intel®	Core™	i7-8665U	
Intel®	Core™	i7-8665UE	
Intel®	Core™	i7-8700	
Intel®	Core™	i7-8700B	
Intel®	Core™	i7-8700K	
Intel®	Core™	i7-8700T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i7-8705G	
Intel®	Core™	i7-8706G	
Intel®	Core™	i7-8709G	
Intel®	Core™	i7-8750H	
Intel®	Core™	i7-8809G	
Intel®	Core™	i7-8850H	
Intel®	Core™	i7-9700	
Intel®	Core™	i7-9700E	
Intel®	Core™	i7-9700F	
Intel®	Core™	i7-9700K	
Intel®	Core™	i7-9700KF	
Intel®	Core™	i7-9700T	
Intel®	Core™	i7-9700TE	
Intel®	Core™	i7-9750H	
Intel®	Core™	i7-9750HF	
Intel®	Core™	i7-9800X	
Intel®	Core™	i7-9850H	
Intel®	Core™	i7-9850HE	
Intel®	Core™	i7-9850HL	
Intel®	Core™	i9-10850K	
Intel®	Core™	i9-10885H	
Intel®	Core™	i9-10900	
Intel®	Core™	i9-10900E	
Intel®	Core™	i9-10900F	
Intel®	Core™	i9-10900K	
Intel®	Core™	i9-10900KF	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-10900T	
Intel®	Core™	i9-10900TE	
Intel®	Core™	i9-10900X	
Intel®	Core™	i9-10920X	
Intel®	Core™	i9-10940X	
Intel®	Core™	i9-10980HK	
Intel®	Core™	i9-10980XE	
Intel®	Core™	i9-11900	
Intel®	Core™	i9-11900F	
Intel®	Core™	i9-11900H	
Intel®	Core™	i9-11900K	
Intel®	Core™	i9-11900KF	
Intel®	Core™	i9-11900T	
Intel®	Core™	i9-11950H	
Intel®	Core™	i9-11980HK	
Intel®	Core™	i9-12900	
Intel®	Core™	i9-12900E	IoT Enterprise Only
Intel®	Core™	i9-12900K	
Intel®	Core™	i9-12900KF	
Intel®	Core™	i9-12900TE	IoT Enterprise Only
Intel®	Core™	i9-13900	
Intel®	Core™	i9-13900E	
Intel®	Core™	i9-13900F	IoT Enterprise Only
Intel®	Core™	i9-13900H	IoT Enterprise Only
Intel®	Core™	i9-13900HK	IoT Enterprise Only
Intel®	Core™	i9-13900HX	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-13900K	
Intel®	Core™	i9-13900KF	IoT Enterprise Only
Intel®	Core™	i9-13900KS	IoT Enterprise Only
Intel®	Core™	i9-13900T	
Intel®	Core™	i9-13900TE	
Intel®	Core™	i9-13905H	IoT Enterprise Only
Intel®	Core™	i9-13950HX	IoT Enterprise Only
Intel®	Core™	i9-13980HX	IoT Enterprise Only
Intel®	Core™	i9-14900	IoT Enterprise Only
Intel®	Core™	i9-14900F	IoT Enterprise Only
Intel®	Core™	i9-14900HX	IoT Enterprise Only
Intel®	Core™	i9-14900K	IoT Enterprise Only
Intel®	Core™	i9-14900KF	IoT Enterprise Only
Intel®	Core™	i9-14900KS	IoT Enterprise Only
Intel®	Core™	i9-14900T	IoT Enterprise Only
Intel®	Core™	i9-7900X	
Intel®	Core™	i9-7920X	
Intel®	Core™	i9-7940X	
Intel®	Core™	i9-7960X	
Intel®	Core™	i9-7980XE	
Intel®	Core™	i9-8950HK	
Intel®	Core™	i9-9820X	
Intel®	Core™	i9-9880H	
Intel®	Core™	i9-9900	
Intel®	Core™	i9-9900K	
Intel®	Core™	i9-9900KF	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	i9-9900KS	
Intel®	Core™	i9-9900T	
Intel®	Core™	i9-9900X	
Intel®	Core™	i9-9920X	
Intel®	Core™	i9-9940X	
Intel®	Core™	i9-9960X	
Intel®	Core™	i9-9980HK	
Intel®	Core™	i9-9980XE	
Intel®	Core™	m3-6Y30	
Intel®	Core™	m3-7Y30	
Intel®	Core™	m3-7Y32	
Intel®	Core™	m3-8100Y	
Intel®	Core™	m5-6Y54	
Intel®	Core™	m5-6Y57	
Intel®	Core™	M-5Y10	
Intel®	Core™	M-5Y10a	
Intel®	Core™	M-5Y10c	
Intel®	Core™	M-5Y31	
Intel®	Core™	M-5Y51	
Intel®	Core™	M-5Y70	
Intel®	Core™	M-5Y71	
Intel®	Core™	m7-6Y75	
Intel®	Core™	Ultra 3 105UL	IoT Enterprise Only
Intel®	Core™	Ultra 5 125H	IoT Enterprise Only
Intel®	Core™	Ultra 5 125HL	IoT Enterprise Only
Intel®	Core™	Ultra 5 125U	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Core™	Ultra 5 125UL	IoT Enterprise Only
Intel®	Core™	Ultra 5 134U	IoT Enterprise Only
Intel®	Core™	Ultra 5 135H	IoT Enterprise Only
Intel®	Core™	Ultra 5 135HL	IoT Enterprise Only
Intel®	Core™	Ultra 5 135U	IoT Enterprise Only
Intel®	Core™	Ultra 7 155H	IoT Enterprise Only
Intel®	Core™	Ultra 7 155HL	IoT Enterprise Only
Intel®	Core™	Ultra 7 155U	IoT Enterprise Only
Intel®	Core™	Ultra 7 155UL	IoT Enterprise Only
Intel®	Core™	Ultra 7 165H	IoT Enterprise Only
Intel®	Core™	Ultra 7 165HL	IoT Enterprise Only
Intel®	Core™	Ultra 7 165U	IoT Enterprise Only
Intel®	Core™	Ultra 9 185H	IoT Enterprise Only
Intel®	Pentium®	6805	
Intel®	Pentium®	3805U	
Intel®	Pentium®	3825U	
Intel®	Pentium®	4405U	
Intel®	Pentium®	4405Y	
Intel®	Pentium®	D1507	
Intel®	Pentium®	D1508	
Intel®	Pentium®	D1509	
Intel®	Pentium®	D1517	
Intel®	Pentium®	D1519	
Intel®	Pentium®	G4400	
Intel®	Pentium®	G4400T	
Intel®	Pentium®	G4400TE	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	G4500	
Intel®	Pentium®	G4500T	
Intel®	Pentium®	G4520	
Intel®	Pentium®	G4560	
Intel®	Pentium®	G4560T	
Intel®	Pentium®	G4600	
Intel®	Pentium®	G4600T	
Intel®	Pentium®	G4620	
Intel®	Pentium®	Gold 4410Y	
Intel®	Pentium®	Gold 4415U	
Intel®	Pentium®	Gold 4415Y	
Intel®	Pentium®	Gold 4417U	
Intel®	Pentium®	Gold 4425Y	
Intel®	Pentium®	Gold 5405U	
Intel®	Pentium®	Gold 6405U	
Intel®	Pentium®	Gold 6500Y	
Intel®	Pentium®	Gold 7505	
Intel®	Pentium®	Gold G5400	
Intel®	Pentium®	Gold G5400T	
Intel®	Pentium®	Gold G5420	
Intel®	Pentium®	Gold G5420T	
Intel®	Pentium®	Gold G5500	
Intel®	Pentium®	Gold G5500T	
Intel®	Pentium®	Gold G5600	
Intel®	Pentium®	Gold G5600T	
Intel®	Pentium®	Gold G5620	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Pentium®	Gold G6400	
Intel®	Pentium®	Gold G6400E	
Intel®	Pentium®	Gold G6400T	
Intel®	Pentium®	Gold G6400TE	
Intel®	Pentium®	Gold G6405	
Intel®	Pentium®	Gold G6405T	
Intel®	Pentium®	Gold G6500	
Intel®	Pentium®	Gold G6500T	
Intel®	Pentium®	Gold G6505	
Intel®	Pentium®	Gold G6505T	
Intel®	Pentium®	Gold G6600	
Intel®	Pentium®	Gold G6605	
Intel®	Pentium®	Gold G7400E	IoT Enterprise Only
Intel®	Pentium®	Gold G7400TE	IoT Enterprise Only
Intel®	Pentium®	J6426	
Intel®	Pentium®	N3710	IoT Enterprise Only
Intel®	Pentium®	N4200	IoT Enterprise Only
Intel®	Pentium®	N6415	
Intel®	Pentium®	Silver J5005	
Intel®	Pentium®	Silver J5040	
Intel®	Pentium®	Silver N5000	
Intel®	Pentium®	Silver N5030	
Intel®	Pentium®	Silver N6000	
Intel®	Pentium®	Silver N6005	
Intel®	Processor	300	IoT Enterprise Only
Intel®	Processor	300T	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Intel®	Processor	N50	IoT Enterprise Only
Intel®	Processor	N95	IoT Enterprise Only
Intel®	Processor	N97	IoT Enterprise Only
Intel®	Processor	N100	IoT Enterprise Only
Intel®	Processor	N200	IoT Enterprise Only
Intel®	Processor	U300	IoT Enterprise Only
Intel®	Processor	U300E	
Intel®	Processor	U300L	IoT Enterprise Only
Intel®	Processor	U301L	IoT Enterprise Only
Intel®	Processor	U302L	IoT Enterprise Only
Intel®	Processor	U303L	IoT Enterprise Only
Intel®	Xeon®	Bronze 3104	
Intel®	Xeon®	Bronze 3106	
Intel®	Xeon®	Bronze 3204	
Intel®	Xeon®	Bronze 3206R	
Intel®	Xeon®	D-1513N	
Intel®	Xeon®	D-1518	
Intel®	Xeon®	D-1520	
Intel®	Xeon®	D-1521	
Intel®	Xeon®	D-1523N	
Intel®	Xeon®	D-1527	
Intel®	Xeon®	D-1528	
Intel®	Xeon®	D-1529	
Intel®	Xeon®	D-1531	
Intel®	Xeon®	D-1533N	
Intel®	Xeon®	D-1537	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	D-1539	
Intel®	Xeon®	D-1540	
Intel®	Xeon®	D-1541	
Intel®	Xeon®	D-1543N	
Intel®	Xeon®	D-1548	
Intel®	Xeon®	D-1553N	
Intel®	Xeon®	D-1557	
Intel®	Xeon®	D-1559	
Intel®	Xeon®	D-1567	
Intel®	Xeon®	D-1571	
Intel®	Xeon®	D-1577	
Intel®	Xeon®	D-2123IT	
Intel®	Xeon®	D-2141I	
Intel®	Xeon®	D-2142IT	
Intel®	Xeon®	D-2143IT	
Intel®	Xeon®	D-2145NT	
Intel®	Xeon®	D-2146NT	
Intel®	Xeon®	D-2161I	
Intel®	Xeon®	D-2163IT	
Intel®	Xeon®	D-2166NT	
Intel®	Xeon®	D-2173IT	
Intel®	Xeon®	D-2177NT	
Intel®	Xeon®	D-2183IT	
Intel®	Xeon®	D-2187NT	
Intel®	Xeon®	E-2124	
Intel®	Xeon®	E-2124G	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2126G	
Intel®	Xeon®	E-2134	
Intel®	Xeon®	E-2136	
Intel®	Xeon®	E-2144G	
Intel®	Xeon®	E-2146G	
Intel®	Xeon®	E-2174G	
Intel®	Xeon®	E-2176G	
Intel®	Xeon®	E-2176M	
Intel®	Xeon®	E-2186G	
Intel®	Xeon®	E-2186M	
Intel®	Xeon®	E-2224	
Intel®	Xeon®	E-2224G	
Intel®	Xeon®	E-2226G	
Intel®	Xeon®	E-2226GE	
Intel®	Xeon®	E-2234	
Intel®	Xeon®	E-2236	
Intel®	Xeon®	E-2244G	
Intel®	Xeon®	E-2246G	
Intel®	Xeon®	E-2254ME	
Intel®	Xeon®	E-2254ML	
Intel®	Xeon®	E-2274G	
Intel®	Xeon®	E-2276G	
Intel®	Xeon®	E-2276M	
Intel®	Xeon®	E-2276ME	
Intel®	Xeon®	E-2276ML	
Intel®	Xeon®	E-2278G	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E-2278GE	
Intel®	Xeon®	E-2278GEL	
Intel®	Xeon®	E-2286G	
Intel®	Xeon®	E-2286M	
Intel®	Xeon®	E-2288G	
Intel®	Xeon®	E-2324G	IoT Enterprise Only
Intel®	Xeon®	E-2414	IoT Enterprise Only
Intel®	Xeon®	E-2434	IoT Enterprise Only
Intel®	Xeon®	E-2436	IoT Enterprise Only
Intel®	Xeon®	E-2456	IoT Enterprise Only
Intel®	Xeon®	E-2468	IoT Enterprise Only
Intel®	Xeon®	E-2478	IoT Enterprise Only
Intel®	Xeon®	E-2486	IoT Enterprise Only
Intel®	Xeon®	E-2488	IoT Enterprise Only
Intel®	Xeon®	E3-1220 v5	
Intel®	Xeon®	E3-1220 v6	
Intel®	Xeon®	E3-1225 v5	
Intel®	Xeon®	E3-1225 v6	
Intel®	Xeon®	E3-1230 v5	
Intel®	Xeon®	E3-1230 v6	
Intel®	Xeon®	E3-1235L v5	
Intel®	Xeon®	E3-1240 v5	
Intel®	Xeon®	E3-1240 v6	
Intel®	Xeon®	E3-1240L v5	
Intel®	Xeon®	E3-1245 v5	
Intel®	Xeon®	E3-1245 v6	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1258L v4	
Intel®	Xeon®	E3-1260L v5	
Intel®	Xeon®	E3-1265L v4	
Intel®	Xeon®	E3-1268L v5	
Intel®	Xeon®	E3-1270 v5	
Intel®	Xeon®	E3-1270 v6	
Intel®	Xeon®	E3-1275 v5	
Intel®	Xeon®	E3-1275 v6	
Intel®	Xeon®	E3-1278L v4	
Intel®	Xeon®	E3-1280 v5	
Intel®	Xeon®	E3-1280 v6	
Intel®	Xeon®	E3-1285 v4	
Intel®	Xeon®	E3-1285 v6	
Intel®	Xeon®	E3-1285L v4	
Intel®	Xeon®	E3-1501L v6	
Intel®	Xeon®	E3-1501M v6	
Intel®	Xeon®	E3-1505L v5	
Intel®	Xeon®	E3-1505L v6	
Intel®	Xeon®	E3-1505M v5	
Intel®	Xeon®	E3-1505M v6	
Intel®	Xeon®	E3-1515M v5	
Intel®	Xeon®	E3-1535M v5	
Intel®	Xeon®	E3-1535M v6	
Intel®	Xeon®	E3-1545M v5	
Intel®	Xeon®	E3-1558L v5	
Intel®	Xeon®	E3-1565L v5	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E3-1575M v5	
Intel®	Xeon®	E3-1578L v5	
Intel®	Xeon®	E3-1585 v5	
Intel®	Xeon®	E3-1585L v5	
Intel®	Xeon®	E5-1620 v4	
Intel®	Xeon®	E5-1630 v4	
Intel®	Xeon®	E5-1650 v4	
Intel®	Xeon®	E5-1660 v4	
Intel®	Xeon®	E5-1680 v4	
Intel®	Xeon®	E5-2603 v4	
Intel®	Xeon®	E5-2608L v4	
Intel®	Xeon®	E5-2609 v4	
Intel®	Xeon®	E5-2618L v4	
Intel®	Xeon®	E5-2620 v4	
Intel®	Xeon®	E5-2623 v4	
Intel®	Xeon®	E5-2628L v4	
Intel®	Xeon®	E5-2630 v4	
Intel®	Xeon®	E5-2630L v4	
Intel®	Xeon®	E5-2637 v4	
Intel®	Xeon®	E5-2640 v4	
Intel®	Xeon®	E5-2643 v4	
Intel®	Xeon®	E5-2648L v4	
Intel®	Xeon®	E5-2650 v4	
Intel®	Xeon®	E5-2650L v4	
Intel®	Xeon®	E5-2658 v4	
Intel®	Xeon®	E5-2660 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E5-2667 v4	
Intel®	Xeon®	E5-2680 v4	
Intel®	Xeon®	E5-2683 v4	
Intel®	Xeon®	E5-2687W v4	
Intel®	Xeon®	E5-2690 v4	
Intel®	Xeon®	E5-2695 v4	
Intel®	Xeon®	E5-2697 v4	
Intel®	Xeon®	E5-2697A v4	
Intel®	Xeon®	E5-2698 v4	
Intel®	Xeon®	E5-2699 v4	
Intel®	Xeon®	E5-2699A v4	
Intel®	Xeon®	E5-2699R v4	
Intel®	Xeon®	E5-4610 v4	
Intel®	Xeon®	E5-4620 v4	
Intel®	Xeon®	E5-4627 v4	
Intel®	Xeon®	E5-4628L v4	
Intel®	Xeon®	E5-4640 v4	
Intel®	Xeon®	E5-4650 v4	
Intel®	Xeon®	E5-4655 v4	
Intel®	Xeon®	E5-4660 v4	
Intel®	Xeon®	E5-4667 v4	
Intel®	Xeon®	E5-4669 v4	
Intel®	Xeon®	E7-4809 v4	
Intel®	Xeon®	E7-4820 v4	
Intel®	Xeon®	E7-4830 v4	
Intel®	Xeon®	E7-4850 v4	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	E7-8860 v4	
Intel®	Xeon®	E7-8867 v4	
Intel®	Xeon®	E7-8870 v4	
Intel®	Xeon®	E7-8880 v4	
Intel®	Xeon®	E7-8890 v4	
Intel®	Xeon®	E7-8891 v4	
Intel®	Xeon®	E7-8893 v4	
Intel®	Xeon®	E7-8894 v4	
Intel®	Xeon®	Gold 5115	
Intel®	Xeon®	Gold 5118	
Intel®	Xeon®	Gold 5119T	
Intel®	Xeon®	Gold 5120	
Intel®	Xeon®	Gold 5120T	
Intel®	Xeon®	Gold 5122	
Intel®	Xeon®	Gold 5215	
Intel®	Xeon®	Gold 5215L	
Intel®	Xeon®	Gold 5217	
Intel®	Xeon®	Gold 5218	
Intel®	Xeon®	Gold 5218B	
Intel®	Xeon®	Gold 5218N	
Intel®	Xeon®	Gold 5218R	
Intel®	Xeon®	Gold 5218T	
Intel®	Xeon®	Gold 5220	
Intel®	Xeon®	Gold 5220R	
Intel®	Xeon®	Gold 5220S	
Intel®	Xeon®	Gold 5220T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 5222	
Intel®	Xeon®	Gold 5315Y	
Intel®	Xeon®	Gold 5317	
Intel®	Xeon®	Gold 5318N	
Intel®	Xeon®	Gold 5318S	
Intel®	Xeon®	Gold 5320	
Intel®	Xeon®	Gold 5320T	
Intel®	Xeon®	Gold 6126	
Intel®	Xeon®	Gold 6126F	
Intel®	Xeon®	Gold 6126T	
Intel®	Xeon®	Gold 6128	
Intel®	Xeon®	Gold 6130	
Intel®	Xeon®	Gold 6130F	
Intel®	Xeon®	Gold 6130T	
Intel®	Xeon®	Gold 6132	
Intel®	Xeon®	Gold 6134	
Intel®	Xeon®	Gold 6136	
Intel®	Xeon®	Gold 6138	
Intel®	Xeon®	Gold 6138F	
Intel®	Xeon®	Gold 6138P	
Intel®	Xeon®	Gold 6138T	
Intel®	Xeon®	Gold 6140	
Intel®	Xeon®	Gold 6142	
Intel®	Xeon®	Gold 6142F	
Intel®	Xeon®	Gold 6144	
Intel®	Xeon®	Gold 6146	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6148	
Intel®	Xeon®	Gold 6148F	
Intel®	Xeon®	Gold 6150	
Intel®	Xeon®	Gold 6152	
Intel®	Xeon®	Gold 6154	
Intel®	Xeon®	Gold 6208U	
Intel®	Xeon®	Gold 6209U	
Intel®	Xeon®	Gold 6210U	
Intel®	Xeon®	Gold 6212U	
Intel®	Xeon®	Gold 6222V	
Intel®	Xeon®	Gold 6226	
Intel®	Xeon®	Gold 6226R	
Intel®	Xeon®	Gold 6230	
Intel®	Xeon®	Gold 6230N	
Intel®	Xeon®	Gold 6230R	
Intel®	Xeon®	Gold 6230T	
Intel®	Xeon®	Gold 6234	
Intel®	Xeon®	Gold 6238	
Intel®	Xeon®	Gold 6238L	
Intel®	Xeon®	Gold 6238R	
Intel®	Xeon®	Gold 6238T	
Intel®	Xeon®	Gold 6240	
Intel®	Xeon®	Gold 6240L	
Intel®	Xeon®	Gold 6240R	
Intel®	Xeon®	Gold 6240Y	
Intel®	Xeon®	Gold 6242	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6242R	
Intel®	Xeon®	Gold 6244	
Intel®	Xeon®	Gold 6246	
Intel®	Xeon®	Gold 6246R	
Intel®	Xeon®	Gold 6248	
Intel®	Xeon®	Gold 6248R	
Intel®	Xeon®	Gold 6250	
Intel®	Xeon®	Gold 6250L	
Intel®	Xeon®	Gold 6252	
Intel®	Xeon®	Gold 6252N	
Intel®	Xeon®	Gold 6254	
Intel®	Xeon®	Gold 6256	
Intel®	Xeon®	Gold 6258R	
Intel®	Xeon®	Gold 6262V	
Intel®	Xeon®	Gold 6312U	
Intel®	Xeon®	Gold 6314U	
Intel®	Xeon®	Gold 6326	
Intel®	Xeon®	Gold 6330	
Intel®	Xeon®	Gold 6330N	
Intel®	Xeon®	Gold 6334	
Intel®	Xeon®	Gold 6336Y	
Intel®	Xeon®	Gold 6338	
Intel®	Xeon®	Gold 6338N	
Intel®	Xeon®	Gold 6338T	
Intel®	Xeon®	Gold 6342	
Intel®	Xeon®	Gold 6346	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Gold 6348	
Intel®	Xeon®	Gold 6354	
Intel®	Xeon®	Gold Gold 5318Y	
Intel®	Xeon®	Platinum 8153	
Intel®	Xeon®	Platinum 8156	
Intel®	Xeon®	Platinum 8158	
Intel®	Xeon®	Platinum 8160	
Intel®	Xeon®	Platinum 8160F	
Intel®	Xeon®	Platinum 8160T	
Intel®	Xeon®	Platinum 8164	
Intel®	Xeon®	Platinum 8168	
Intel®	Xeon®	Platinum 8170	
Intel®	Xeon®	Platinum 8176	
Intel®	Xeon®	Platinum 8176F	
Intel®	Xeon®	Platinum 8180	
Intel®	Xeon®	Platinum 8253	
Intel®	Xeon®	Platinum 8256	
Intel®	Xeon®	Platinum 8260	
Intel®	Xeon®	Platinum 8260L	
Intel®	Xeon®	Platinum 8260Y	
Intel®	Xeon®	Platinum 8268	
Intel®	Xeon®	Platinum 8270	
Intel®	Xeon®	Platinum 8276	
Intel®	Xeon®	Platinum 8276L	
Intel®	Xeon®	Platinum 8280	
Intel®	Xeon®	Platinum 8280L	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Platinum 8351N	
Intel®	Xeon®	Platinum 8352M	
Intel®	Xeon®	Platinum 8352S	
Intel®	Xeon®	Platinum 8352V	
Intel®	Xeon®	Platinum 8352Y	
Intel®	Xeon®	Platinum 8358	
Intel®	Xeon®	Platinum 8358P	
Intel®	Xeon®	Platinum 8360Y	
Intel®	Xeon®	Platinum 8362	
Intel®	Xeon®	Platinum 8368	
Intel®	Xeon®	Platinum 8368Q	
Intel®	Xeon®	Platinum 8380	
Intel®	Xeon®	Platinum 9221	
Intel®	Xeon®	Platinum 9222	
Intel®	Xeon®	Platinum 9242	
Intel®	Xeon®	Platinum 9282	
Intel®	Xeon®	Silver 4108	
Intel®	Xeon®	Silver 4109T	
Intel®	Xeon®	Silver 4110	
Intel®	Xeon®	Silver 4112	
Intel®	Xeon®	Silver 4114	
Intel®	Xeon®	Silver 4114T	
Intel®	Xeon®	Silver 4116	
Intel®	Xeon®	Silver 4116T	
Intel®	Xeon®	Silver 4208	
Intel®	Xeon®	Silver 4209T	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	Silver 4210	
Intel®	Xeon®	Silver 4210R	
Intel®	Xeon®	Silver 4210T	
Intel®	Xeon®	Silver 4214	
Intel®	Xeon®	Silver 4214R	
Intel®	Xeon®	Silver 4214Y	
Intel®	Xeon®	Silver 4215	
Intel®	Xeon®	Silver 4215R	
Intel®	Xeon®	Silver 4216	
Intel®	Xeon®	Silver 4309Y	
Intel®	Xeon®	Silver 4310	
Intel®	Xeon®	Silver 4310T	
Intel®	Xeon®	Silver 4314	
Intel®	Xeon®	Silver 4316	
Intel®	Xeon®	W-10855M	
Intel®	Xeon®	W-10885M	
Intel®	Xeon®	W-11855M	
Intel®	Xeon®	W-11955M	
Intel®	Xeon®	W-1250	
Intel®	Xeon®	W-1250E	
Intel®	Xeon®	W-1250P	
Intel®	Xeon®	W-1250TE	
Intel®	Xeon®	W-1270	
Intel®	Xeon®	W-1270E	
Intel®	Xeon®	W-1270P	
Intel®	Xeon®	W-1270TE	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-1290	
Intel®	Xeon®	W-1290E	
Intel®	Xeon®	W-1290P	
Intel®	Xeon®	W-1290T	
Intel®	Xeon®	W-1290TE	
Intel®	Xeon®	W-2102	
Intel®	Xeon®	W-2104	
Intel®	Xeon®	W-2123	
Intel®	Xeon®	W-2125	
Intel®	Xeon®	W-2133	
Intel®	Xeon®	W-2135	
Intel®	Xeon®	W-2145	
Intel®	Xeon®	W-2155	
Intel®	Xeon®	W-2175	
Intel®	Xeon®	W-2195	
Intel®	Xeon®	W-2223	
Intel®	Xeon®	W-2225	
Intel®	Xeon®	W-2235	
Intel®	Xeon®	W-2245	
Intel®	Xeon®	W-2255	
Intel®	Xeon®	W-2265	
Intel®	Xeon®	W-2275	
Intel®	Xeon®	W-2295	
Intel®	Xeon®	W-3175X	
Intel®	Xeon®	W-3223	
Intel®	Xeon®	W-3225	

Manufacturer	Brand	Model	IoT Applicability
Intel®	Xeon®	W-3235	
Intel®	Xeon®	W-3245	
Intel®	Xeon®	W-3245M	
Intel®	Xeon®	W-3265	
Intel®	Xeon®	W-3265M	
Intel®	Xeon®	W-3275	
Intel®	Xeon®	W-3275M	
Intel®	Xeon®	W3-2423	
Intel®	Xeon®	W3-2425	
Intel®	Xeon®	W3-2435	
Intel®	Xeon®	W5-2445	
Intel®	Xeon®	W5-2455X	
Intel®	Xeon®	W5-2465X	
Intel®	Xeon®	W5-3423	
Intel®	Xeon®	W5-3425	
Intel®	Xeon®	W5-3433	
Intel®	Xeon®	W5-3435X	
Intel®	Xeon®	W7-2475X	
Intel®	Xeon®	W7-2495X	
Intel®	Xeon®	W7-3445	
Intel®	Xeon®	W7-3455	
Intel®	Xeon®	W7-3465X	
Intel®	Xeon®	W9-3475X	
Intel®	Xeon®	W9-3495X	

Windows 11 version 22H2 & 23H2 supported AMD processors

Article • 03/01/2024

The processors listed represent the processor models which meet the minimum floor for the supported processor generations and up through the latest processors at the time of publication. These processors meet the design principles around security, reliability, and the minimum system requirements for Windows 11. Subsequently released and future generations of processors which meet the same principles will be considered as supported, even if not explicitly listed. It is expected that the processor list will not reflect the most current offerings from processor manufacturers between updates. Updates to the processor lists will occur at each subsequent general availability of Windows.

OEMs may use the following CPUs for new Windows 11 devices. New Windows 11 devices must use modern device drivers which have passed the Windows Hardware Compatibility Program for Windows 11 or the latest available modern device drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

Expand table

Manufacturer	Brand	Model
AMD	AMD	3015e
AMD	AMD	3020e
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	7120e
AMD	Athlon™	7120U
AMD	Athlon™	7220e
AMD	Athlon™	7220U
AMD	Athlon™	Gold 3150C

Manufacturer	Brand	Model
AMD	Athlon™	Gold 3150G
AMD	Athlon™	Gold 3150GE
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE
AMD	Athlon™	Silver 3050U
AMD	Athlon™	Gold PRO 3125GE
AMD	Athlon™	Gold PRO 3150G
AMD	Athlon™	Gold PRO 3150GE
AMD	Athlon™	Gold PRO 4150GE
AMD	Athlon™	PRO 300GE
AMD	Athlon™	PRO 300U
AMD	Athlon™	PRO 3045B
AMD	EPYC™	7252
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7282
AMD	EPYC™	7302
AMD	EPYC™	7313
AMD	EPYC™	7343
AMD	EPYC™	7352
AMD	EPYC™	7402
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7452

Manufacturer	Brand	Model
AMD	EPYC™	7453
AMD	EPYC™	7502
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7552
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7713
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7232P
AMD	EPYC™	72F3
AMD	EPYC™	7302P
AMD	EPYC™	7313P
AMD	EPYC™	73F3
AMD	EPYC™	7402P
AMD	EPYC™	7443P
AMD	EPYC™	74F3
AMD	EPYC™	7502P
AMD	EPYC™	7543P
AMD	EPYC™	75F3

Manufacturer	Brand	Model
AMD	EPYC™	7702P
AMD	EPYC™	7713P
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	EPYC™	8024P
AMD	EPYC™	8024PN
AMD	EPYC™	8124P
AMD	EPYC™	8124PN
AMD	EPYC™	8224P
AMD	EPYC™	8224PN
AMD	EPYC™	8324P
AMD	EPYC™	8324PN
AMD	EPYC™	8434P
AMD	EPYC™	8434PN
AMD	EPYC™	8534P
AMD	EPYC™	8534PN
AMD	EPYC™	9124
AMD	EPYC™	9174F
AMD	EPYC™	9184X
AMD	EPYC™	9224
AMD	EPYC™	9254
AMD	EPYC™	9274F
AMD	EPYC™	9334
AMD	EPYC™	9354

Manufacturer	Brand	Model
AMD	EPYC™	9354P
AMD	EPYC™	9374F
AMD	EPYC™	9384X
AMD	EPYC™	9454
AMD	EPYC™	9454P
AMD	EPYC™	9474F
AMD	EPYC™	9534
AMD	EPYC™	9554
AMD	EPYC™	9554P
AMD	EPYC™	9634
AMD	EPYC™	9654
AMD	EPYC™	9654P
AMD	EPYC™	9684X
AMD	EPYC™	9734
AMD	EPYC™	9754
AMD	EPYC™	9754S
AMD	Ryzen Embedded R2000 Series	R2312
AMD	Ryzen Embedded R2000 Series	R2314
AMD	Ryzen Embedded R2000 Series	R2514
AMD	Ryzen Embedded R2000 Series	R2544
AMD	Ryzen™	Z1
AMD	Ryzen™	Z1 Extreme
AMD	Ryzen™ 3	3100
AMD	Ryzen™ 3	4100
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics

Manufacturer	Brand	Model
AMD	Ryzen™ 3	3200GE
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G
AMD	Ryzen™ 3	4300GE
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3	5125C
AMD	Ryzen™ 3	5300G (OEM Only)
AMD	Ryzen™ 3	5300GE
AMD	Ryzen™ 3	5300GE (OEM Only)
AMD	Ryzen™ 3	5300U
AMD	Ryzen™ 3	5400U
AMD	Ryzen™ 3	5425C
AMD	Ryzen™ 3	5425U
AMD	Ryzen™ 3	7320e
AMD	Ryzen™ 3	7320U
AMD	Ryzen™ 3	7330U
AMD	Ryzen™ 3	7335U
AMD	Ryzen™ 3	7440U
AMD	Ryzen™ 3	5380U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U

Manufacturer	Brand	Model
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 3 PRO	5350G
AMD	Ryzen™ 3 PRO	5350GE
AMD	Ryzen™ 3 PRO	5450U
AMD	Ryzen™ 3 PRO	5475U
AMD	Ryzen™ 3 PRO	7330U
AMD	Ryzen™ 3 PRO	4355G
AMD	Ryzen™ 3 PRO	4355GE
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	3600
AMD	Ryzen™ 5	4500
AMD	Ryzen™ 5	5500
AMD	Ryzen™ 5	5600
AMD	Ryzen™ 5	7600
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3350G
AMD	Ryzen™ 5	3350GE
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor
AMD	Ryzen™ 5	3500C

Manufacturer	Brand	Model
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3500X
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G
AMD	Ryzen™ 5	4600GE
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600HS
AMD	Ryzen™ 5	4600U
AMD	Ryzen™ 5	5300G
AMD	Ryzen™ 5	5300GE
AMD	Ryzen™ 5	5425U
AMD	Ryzen™ 5	5500U
AMD	Ryzen™ 5	5560U
AMD	Ryzen™ 5	5600G
AMD	Ryzen™ 5	5600GE
AMD	Ryzen™ 5	5600H
AMD	Ryzen™ 5	5600HS
AMD	Ryzen™ 5	5600U
AMD	Ryzen™ 5	5600X
AMD	Ryzen™ 5	5625C
AMD	Ryzen™ 5	5625U
AMD	Ryzen™ 5	6600H

Manufacturer	Brand	Model
AMD	Ryzen™ 5	6600HS
AMD	Ryzen™ 5	6600U
AMD	Ryzen™ 5	7520U
AMD	Ryzen™ 5	7530U
AMD	Ryzen™ 5	7535HS
AMD	Ryzen™ 5	7535U
AMD	Ryzen™ 5	7540U
AMD	Ryzen™ 5	7600X
AMD	Ryzen™ 5	7640HS
AMD	Ryzen™ 5	7640S
AMD	Ryzen™ 5	7640U
AMD	Ryzen™ 5	7645HX
AMD	Ryzen™ 5	7640H
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	5645
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 5 PRO	5475U
AMD	Ryzen™ 5 PRO	5650G

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	5650GE
AMD	Ryzen™ 5 PRO	5650HS
AMD	Ryzen™ 5 PRO	5650HX
AMD	Ryzen™ 5 PRO	5650U
AMD	Ryzen™ 5 PRO	5675U
AMD	Ryzen™ 5 PRO	5750G
AMD	Ryzen™ 5 PRO	5750GE
AMD	Ryzen™ 5 PRO	6650H
AMD	Ryzen™ 5 PRO	6650HS
AMD	Ryzen™ 5 PRO	6650U
AMD	Ryzen™ 5 PRO	7530U
AMD	Ryzen™ 5 PRO	7540U
AMD	Ryzen™ 5 PRO	7640U
AMD	Ryzen™ 5 PRO	4655G
AMD	Ryzen™ 5 PRO	4655GE
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	7700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition

Manufacturer	Brand	Model
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G
AMD	Ryzen™ 7	4700GE
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7	5700G
AMD	Ryzen™ 7	5700GE
AMD	Ryzen™ 7	5700U
AMD	Ryzen™ 7	5700X
AMD	Ryzen™ 7	5800H
AMD	Ryzen™ 7	5800HS
AMD	Ryzen™ 7	5800U
AMD	Ryzen™ 7	5800X
AMD	Ryzen™ 7	5800X3D
AMD	Ryzen™ 7	5825C
AMD	Ryzen™ 7	5825U
AMD	Ryzen™ 7	6800H
AMD	Ryzen™ 7	6800HS
AMD	Ryzen™ 7	6800U
AMD	Ryzen™ 7	6810U
AMD	Ryzen™ 7	7700X
AMD	Ryzen™ 7	7730U
AMD	Ryzen™ 7	7735HS

Manufacturer	Brand	Model
AMD	Ryzen™ 7	7735U
AMD	Ryzen™ 7	7736U
AMD	Ryzen™ 7	7745HX
AMD	Ryzen™ 7	7800X3D
AMD	Ryzen™ 7	7840H
AMD	Ryzen™ 7	7840HS
AMD	Ryzen™ 7	7840S
AMD	Ryzen™ 7	7840U
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	5845
AMD	Ryzen™ 7 PRO	2700X
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 7 PRO	5850HS
AMD	Ryzen™ 7 PRO	5850HX
AMD	Ryzen™ 7 PRO	5850U
AMD	Ryzen™ 7 PRO	5875U
AMD	Ryzen™ 7 PRO	6850H
AMD	Ryzen™ 7 PRO	6850HS
AMD	Ryzen™ 7 PRO	6850U
AMD	Ryzen™ 7 PRO	6860Z
AMD	Ryzen™ 7 PRO	7730U
AMD	Ryzen™ 7 PRO	7840U

Manufacturer	Brand	Model
AMD	Ryzen™ 9	5900
AMD	Ryzen™ 9	7900
AMD	Ryzen™ 9	3900 Processor
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9	5900HS
AMD	Ryzen™ 9	5900HX
AMD	Ryzen™ 9	5900X
AMD	Ryzen™ 9	5950X
AMD	Ryzen™ 9	5980HS
AMD	Ryzen™ 9	5980HX
AMD	Ryzen™ 9	6900HS
AMD	Ryzen™ 9	6900HX
AMD	Ryzen™ 9	6980HS
AMD	Ryzen™ 9	6980HX
AMD	Ryzen™ 9	7845HX
AMD	Ryzen™ 9	7900X
AMD	Ryzen™ 9	7900X3D
AMD	Ryzen™ 9	7940H
AMD	Ryzen™ 9	7940HS
AMD	Ryzen™ 9	7945HX
AMD	Ryzen™ 9	7950X
AMD	Ryzen™ 9	7950X3D

Manufacturer	Brand	Model
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ 9 PRO	5945
AMD	Ryzen™ 9 PRO	6950H
AMD	Ryzen™ 9 PRO	6950HS
AMD	Ryzen™ Embedded	V2516
AMD	Ryzen™ Embedded	V2546
AMD	Ryzen™ Embedded	V2718
AMD	Ryzen™ Embedded	V2748
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX
AMD	Ryzen™ Threadripper™ PRO	5945WX
AMD	Ryzen™ Threadripper™ PRO	5955WX
AMD	Ryzen™ Threadripper™ PRO	5965WX
AMD	Ryzen™ Threadripper™ PRO	5975WX
AMD	Ryzen™ Threadripper™ PRO	5995WX

Windows 11 version 22H2 & 23H2 supported Intel processors

Article • 11/30/2023

The processors listed represent the processor models which meet the minimum floor for the supported processor generations and up through the latest processors at the time of publication. These processors meet the design principles around security, reliability, and the minimum system requirements for Windows 11. Subsequently released and future generations of processors which meet the same principles will be considered as supported, even if not explicitly listed. It is expected that the processor list will not reflect the most current offerings from processor manufacturers between updates. Updates to the processor lists will occur at each subsequent general availability of Windows.

OEMs may use the following CPUs for new Windows 11 devices. New Windows 11 devices must use modern device drivers which have passed the Windows Hardware Compatibility Program for Windows 11 or the latest available modern device drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

Expand table

Manufacturer	Brand	Model
Intel®	Atom®	x6200FE
Intel®	Atom®	x6211E
Intel®	Atom®	x6212RE
Intel®	Atom®	x6413E
Intel®	Atom®	x6414RE
Intel®	Atom®	x6425E
Intel®	Atom®	x6425RE
Intel®	Atom®	x6427FE
Intel®	Celeron®	6305
Intel®	Celeron®	7300
Intel®	Celeron®	7305

Manufacturer	Brand	Model
Intel®	Celeron®	3867U
Intel®	Celeron®	4205U
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	6305E
Intel®	Celeron®	6600HE
Intel®	Celeron®	7305E
Intel®	Celeron®	7305L
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925

Manufacturer	Brand	Model
Intel®	Celeron®	G6900
Intel®	Celeron®	G6900E
Intel®	Celeron®	G6900T
Intel®	Celeron®	G6900TE
Intel®	Celeron®	J4005[2]
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105[2]
Intel®	Celeron®	J4115[2]
Intel®	Celeron®	J4125
Intel®	Celeron®	J6412
Intel®	Celeron®	J6413
Intel®	Celeron®	N4000[2]
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100[2]
Intel®	Celeron®	N4120
Intel®	Celeron®	N4500
Intel®	Celeron®	N4505
Intel®	Celeron®	N5100
Intel®	Celeron®	N5105
Intel®	Celeron®	N6210
Intel®	Celeron®	N6211
Intel®	Core™	3 processor 100U
Intel®	Core™	5 processor 120U
Intel®	Core™	7 processor 150U
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4

Manufacturer	Brand	Model
Intel®	Core™	i3-1005G1
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-11100HE
Intel®	Core™	i3-1110G4
Intel®	Core™	i3-1115G4
Intel®	Core™	i3-1115G4E
Intel®	Core™	i3-1115GRE
Intel®	Core™	i3-1120G4
Intel®	Core™	i3-1125G4
Intel®	Core™	i3-12100

Manufacturer	Brand	Model
Intel®	Core™	i3-12100E
Intel®	Core™	i3-12100F
Intel®	Core™	i3-12100T
Intel®	Core™	i3-12100TE
Intel®	Core™	i3-1210U
Intel®	Core™	i3-1215U
Intel®	Core™	i3-1215UE
Intel®	Core™	i3-1215UL
Intel®	Core™	i3-1220P
Intel®	Core™	i3-1220PE
Intel®	Core™	i3-12300
Intel®	Core™	i3-12300HE
Intel®	Core™	i3-12300HL
Intel®	Core™	i3-12300T
Intel®	Core™	i3-1305U
Intel®	Core™	i3-13100
Intel®	Core™	i3-13100E
Intel®	Core™	i3-13100F
Intel®	Core™	i3-13100T
Intel®	Core™	i3-13100TE
Intel®	Core™	i3-1315U
Intel®	Core™	i3-1315UE
Intel®	Core™	i3-1320PE
Intel®	Core™	i3-13300HE
Intel®	Core™	i3-14100
Intel®	Core™	i3-14100F

Manufacturer	Brand	Model
Intel®	Core™	i3-14100T
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i3-N300
Intel®	Core™	i3-N305

Manufacturer	Brand	Model
Intel®	Core™	i5-10200H
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T
Intel®	Core™	i5-10500TE
Intel®	Core™	i5-10505
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T

Manufacturer	Brand	Model
Intel®	Core™	i5-11260H
Intel®	Core™	i5-11300H
Intel®	Core™	i5-1130G7
Intel®	Core™	i5-11320H
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-11400
Intel®	Core™	i5-11400F
Intel®	Core™	i5-11400H
Intel®	Core™	i5-11400T
Intel®	Core™	i5-1140G7
Intel®	Core™	i5-1145G7
Intel®	Core™	i5-1145G7E
Intel®	Core™	i5-1145GRE
Intel®	Core™	i5-11500
Intel®	Core™	i5-11500H
Intel®	Core™	i5-11500HE
Intel®	Core™	i5-11500T
Intel®	Core™	i5-1155G7
Intel®	Core™	i5-11600
Intel®	Core™	i5-11600K
Intel®	Core™	i5-11600KF
Intel®	Core™	i5-11600T
Intel®	Core™	i5-1230U
Intel®	Core™	i5-1235U
Intel®	Core™	i5-1235UL
Intel®	Core™	i5-12400

Manufacturer	Brand	Model
Intel®	Core™	i5-12400F
Intel®	Core™	i5-12400T
Intel®	Core™	i5-1240P
Intel®	Core™	i5-1240U
Intel®	Core™	i5-12450H
Intel®	Core™	i5-12450HX
Intel®	Core™	i5-1245U
Intel®	Core™	i5-1245UE
Intel®	Core™	i5-1245UL
Intel®	Core™	i5-12500
Intel®	Core™	i5-12500E
Intel®	Core™	i5-12500H
Intel®	Core™	i5-12500HL
Intel®	Core™	i5-12500T
Intel®	Core™	i5-12500TE
Intel®	Core™	i5-1250P
Intel®	Core™	i5-1250PE
Intel®	Core™	i5-12600
Intel®	Core™	i5-12600H
Intel®	Core™	i5-12600HE
Intel®	Core™	i5-12600HL
Intel®	Core™	i5-12600HX
Intel®	Core™	i5-12600K
Intel®	Core™	i5-12600KF
Intel®	Core™	i5-12600T
Intel®	Core™	i5-1334U

Manufacturer	Brand	Model
Intel®	Core™	i5-1335U
Intel®	Core™	i5-1335UE
Intel®	Core™	i5-13400
Intel®	Core™	i5-13400E
Intel®	Core™	i5-13400F
Intel®	Core™	i5-13400T
Intel®	Core™	i5-1340P
Intel®	Core™	i5-1340PE
Intel®	Core™	i5-13420H
Intel®	Core™	i5-13450HX
Intel®	Core™	i5-1345U
Intel®	Core™	i5-1345UE
Intel®	Core™	i5-13490F
Intel®	Core™	i5-13500
Intel®	Core™	i5-13500E
Intel®	Core™	i5-13500H
Intel®	Core™	i5-13500HX
Intel®	Core™	i5-13500T
Intel®	Core™	i5-13500TE
Intel®	Core™	i5-13505H
Intel®	Core™	i5-1350P
Intel®	Core™	i5-1350PE
Intel®	Core™	i5-13600
Intel®	Core™	i5-13600H
Intel®	Core™	i5-13600HE
Intel®	Core™	i5-13600HX

Manufacturer	Brand	Model
Intel®	Core™	i5-13600K
Intel®	Core™	i5-13600KF
Intel®	Core™	i5-13600T
Intel®	Core™	i5-14400
Intel®	Core™	i5-14400F
Intel®	Core™	i5-14400T
Intel®	Core™	i5-14450HX
Intel®	Core™	i5-14490F
Intel®	Core™	i5-14500
Intel®	Core™	i5-14500HX
Intel®	Core™	i5-14500T
Intel®	Core™	i5-14600
Intel®	Core™	i5-14600K
Intel®	Core™	i5-14600KF
Intel®	Core™	i5-14600T
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8305G

Manufacturer	Brand	Model
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600

Manufacturer	Brand	Model
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i7-10510U
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-11370H
Intel®	Core™	i7-11375H
Intel®	Core™	i7-11390H

Manufacturer	Brand	Model
Intel®	Core™	i7-11600H
Intel®	Core™	i7-1160G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-11700
Intel®	Core™	i7-11700F
Intel®	Core™	i7-11700K
Intel®	Core™	i7-11700KF
Intel®	Core™	i7-11700T
Intel®	Core™	i7-11800H
Intel®	Core™	i7-1180G7
Intel®	Core™	i7-11850H
Intel®	Core™	i7-11850HE
Intel®	Core™	i7-1185G7
Intel®	Core™	i7-1185G7E
Intel®	Core™	i7-1185GRE
Intel®	Core™	i7-1195G7
Intel®	Core™	i7-1250U
Intel®	Core™	i7-1255U
Intel®	Core™	i7-1255UL
Intel®	Core™	i7-1260P
Intel®	Core™	i7-1260U
Intel®	Core™	i7-12650H
Intel®	Core™	i7-12650HX
Intel®	Core™	i7-1265U
Intel®	Core™	i7-1265UE
Intel®	Core™	i7-1265UL

Manufacturer	Brand	Model
Intel®	Core™	i7-12700
Intel®	Core™	i7-12700E
Intel®	Core™	i7-12700F
Intel®	Core™	i7-12700H
Intel®	Core™	i7-12700HL
Intel®	Core™	i7-12700K
Intel®	Core™	i7-12700KF
Intel®	Core™	i7-12700T
Intel®	Core™	i7-12700TE
Intel®	Core™	i7-1270P
Intel®	Core™	i7-1270PE
Intel®	Core™	i7-12800H
Intel®	Core™	i7-12800HE
Intel®	Core™	i7-12800HL
Intel®	Core™	i7-12800HX
Intel®	Core™	i7-1280P
Intel®	Core™	i7-12850HX
Intel®	Core™	i7-1355U
Intel®	Core™	i7-1360P
Intel®	Core™	i7-13620H
Intel®	Core™	i7-13650HX
Intel®	Core™	i7-1365U
Intel®	Core™	i7-1365UE
Intel®	Core™	i7-13700
Intel®	Core™	i7-13700E
Intel®	Core™	i7-13700F

Manufacturer	Brand	Model
Intel®	Core™	i7-13700H
Intel®	Core™	i7-13700HX
Intel®	Core™	i7-13700K
Intel®	Core™	i7-13700KF
Intel®	Core™	i7-13700T
Intel®	Core™	i7-13700TE
Intel®	Core™	i7-13705H
Intel®	Core™	i7-1370P
Intel®	Core™	i7-1370PE
Intel®	Core™	i7-13790F
Intel®	Core™	i7-13800H
Intel®	Core™	i7-13800HE
Intel®	Core™	i7-13850HX
Intel®	Core™	i7-14650HX
Intel®	Core™	i7-14700
Intel®	Core™	i7-14700F
Intel®	Core™	i7-14700HX
Intel®	Core™	i7-14700K
Intel®	Core™	i7-14700KF
Intel®	Core™	i7-14700T
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820HQ[1]
Intel®	Core™	i7-7820X
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U

Manufacturer	Brand	Model
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8705G
Intel®	Core™	i7-8706G
Intel®	Core™	i7-8709G
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8809G
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF

Manufacturer	Brand	Model
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-11900
Intel®	Core™	i9-11900F
Intel®	Core™	i9-11900H
Intel®	Core™	i9-11900K
Intel®	Core™	i9-11900KF
Intel®	Core™	i9-11900T
Intel®	Core™	i9-11950H
Intel®	Core™	i9-11980HK

Manufacturer	Brand	Model
Intel®	Core™	i9-12900
Intel®	Core™	i9-12900E
Intel®	Core™	i9-12900F
Intel®	Core™	i9-12900H
Intel®	Core™	i9-12900HK
Intel®	Core™	i9-12900HX
Intel®	Core™	i9-12900K
Intel®	Core™	i9-12900KF
Intel®	Core™	i9-12900KS
Intel®	Core™	i9-12900T
Intel®	Core™	i9-12900TE
Intel®	Core™	i9-12950HX
Intel®	Core™	i9-13900
Intel®	Core™	i9-13900E
Intel®	Core™	i9-13900F
Intel®	Core™	i9-13900H
Intel®	Core™	i9-13900HK
Intel®	Core™	i9-13900HX
Intel®	Core™	i9-13900K
Intel®	Core™	i9-13900KF
Intel®	Core™	i9-13900KS
Intel®	Core™	i9-13900T
Intel®	Core™	i9-13900TE
Intel®	Core™	i9-13905H
Intel®	Core™	i9-13950HX
Intel®	Core™	i9-13980HX

Manufacturer	Brand	Model
Intel®	Core™	i9-14900
Intel®	Core™	i9-14900F
Intel®	Core™	i9-14900HX
Intel®	Core™	i9-14900K
Intel®	Core™	i9-14900KF
Intel®	Core™	i9-14900T
Intel®	Core™	i9-7900X
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-8100Y

Manufacturer	Brand	Model
Intel®	Core™	Ultra 5 processor 125H
Intel®	Core™	Ultra 5 processor 125U
Intel®	Core™	Ultra 5 processor 134U
Intel®	Core™	Ultra 5 processor 135H
Intel®	Core™	Ultra 5 processor 135U
Intel®	Core™	Ultra 7 processor 155H
Intel®	Core™	Ultra 7 processor 155U
Intel®	Core™	Ultra 7 processor 164U
Intel®	Core™	Ultra 7 processor 165H
Intel®	Core™	Ultra 7 processor 165U
Intel®	Core™	Ultra 9 processor 185H
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold 6805
Intel®	Pentium®	Gold 7505
Intel®	Pentium®	Gold 8500
Intel®	Pentium®	Gold 8505
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600E
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	Gold G7400
Intel®	Pentium®	Gold G7400E
Intel®	Pentium®	Gold G7400T
Intel®	Pentium®	Gold G7400TE
Intel®	Pentium®	J6426
Intel®	Pentium®	N6415
Intel®	Pentium®	Silver J5005[2]
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000[2]
Intel®	Pentium®	Silver N5030

Manufacturer	Brand	Model
Intel®	Pentium®	Silver N6000
Intel®	Pentium®	Silver N6005
Intel®	Processor	N100
Intel®	Processor	N200
Intel®	Processor	N50
Intel®	Processor	N95
Intel®	Processor	N97
Intel®	Processor	U300
Intel®	Processor	U300E
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1702
Intel®	Xeon®	D-1712TR
Intel®	Xeon®	D-1713NT
Intel®	Xeon®	D-1713NTE
Intel®	Xeon®	D-1714
Intel®	Xeon®	D-1715TER
Intel®	Xeon®	D-1718T
Intel®	Xeon®	D-1722NE
Intel®	Xeon®	D-1726
Intel®	Xeon®	D-1732TE
Intel®	Xeon®	D-1733NT
Intel®	Xeon®	D-1735TR
Intel®	Xeon®	D-1736

Manufacturer	Brand	Model
Intel®	Xeon®	D-1736NT
Intel®	Xeon®	D-1739
Intel®	Xeon®	D-1746TER
Intel®	Xeon®	D-1747NTE
Intel®	Xeon®	D-1748TE
Intel®	Xeon®	D-1749NT
Intel®	Xeon®	D-2712T
Intel®	Xeon®	D-2733NT
Intel®	Xeon®	D-2738
Intel®	Xeon®	D-2752NTE
Intel®	Xeon®	D-2752TER
Intel®	Xeon®	D-2753NT
Intel®	Xeon®	D-2766NT
Intel®	Xeon®	D-2775TE
Intel®	Xeon®	D-2776NT
Intel®	Xeon®	D-2779
Intel®	Xeon®	D-2786NTE
Intel®	Xeon®	D-2795NT
Intel®	Xeon®	D-2796NT
Intel®	Xeon®	D-2796TE
Intel®	Xeon®	D-2798NT
Intel®	Xeon®	D-2799
Intel®	Xeon®	E-2104G
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G

Manufacturer	Brand	Model
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE

Manufacturer	Brand	Model
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5215M
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5318Y
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 5415+
Intel®	Xeon®	Gold 5416S
Intel®	Xeon®	Gold 5418Y
Intel®	Xeon®	Gold 5420+
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6234
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238M
Intel®	Xeon®	Gold 6238R
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6240M
Intel®	Xeon®	Gold 6240R
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold 6416H
Intel®	Xeon®	Gold 6418H
Intel®	Xeon®	Gold 6430
Intel®	Xeon®	Gold 6438Y+
Intel®	Xeon®	Gold 6442Y
Intel®	Xeon®	Gold 6444Y
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8171M
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260M
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8272CL
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8276M
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8280M
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352M
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8362
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8452Y
Intel®	Xeon®	Platinum 8460Y+
Intel®	Xeon®	Platinum 8468
Intel®	Xeon®	Platinum 8470
Intel®	Xeon®	Platinum 8480+
Intel®	Xeon®	Platinum 8490H
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316
Intel®	Xeon®	Silver 4410T
Intel®	Xeon®	Silver 4410Y
Intel®	Xeon®	Silver 4416+
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-11055M
Intel®	Xeon®	W-11155MLE
Intel®	Xeon®	W-11155MRE
Intel®	Xeon®	W-11555MLE
Intel®	Xeon®	W-11555MRE
Intel®	Xeon®	W-11855M
Intel®	Xeon®	W-11865MLE
Intel®	Xeon®	W-11865MRE
Intel®	Xeon®	W-11955M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE

Manufacturer	Brand	Model
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-1350
Intel®	Xeon®	W-1350P
Intel®	Xeon®	W-1370
Intel®	Xeon®	W-1370P
Intel®	Xeon®	W-1390
Intel®	Xeon®	W-1390P
Intel®	Xeon®	W-1390T
Intel®	Xeon®	W-2102
Intel®	Xeon®	W-2104
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195

Manufacturer	Brand	Model
Intel®	Xeon®	W-2223
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235
Intel®	Xeon®	W3-2423
Intel®	Xeon®	W3-2425
Intel®	Xeon®	W3-2435
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M
Intel®	Xeon®	W-3323
Intel®	Xeon®	W-3335
Intel®	Xeon®	W-3345
Intel®	Xeon®	W-3365
Intel®	Xeon®	W-3375

Manufacturer	Brand	Model
Intel®	Xeon®	W5-2445
Intel®	Xeon®	W5-2455X
Intel®	Xeon®	W5-2465X
Intel®	Xeon®	W5-3423
Intel®	Xeon®	W5-3425
Intel®	Xeon®	W5-3433
Intel®	Xeon®	W5-3435X
Intel®	Xeon®	W7-2475X
Intel®	Xeon®	W7-2495X
Intel®	Xeon®	W7-3445
Intel®	Xeon®	W7-3455
Intel®	Xeon®	W7-3465X
Intel®	Xeon®	W9-3475X
Intel®	Xeon®	W9-3495X

[1] Only select devices that shipped with modern drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

[2] Devices with these CPUs may not be manufactured with Windows 11 pre-installed and may only be upgraded to Windows 11 by a customer.

Windows 11, version 22H2 & 23H2 supported Qualcomm processors

Article • 11/30/2023

The processors listed represent the processor models which meet the minimum floor for the supported processor generations and up through the latest processors at the time of publication. These processors meet the design principles around security, reliability, and the minimum system requirements for Windows 11. Subsequently released and future generations of processors which meet the same principles will be considered as supported, even if not explicitly listed. It is expected that the processor list will not reflect the most current offerings from processor manufacturers between updates. Updates to the processor lists will occur at each subsequent general availability of Windows.

OEMs may use the following CPUs for new Windows 11 devices. New Windows 11 devices must use modern device drivers which have passed the Windows Hardware Compatibility Program for Windows 11 or the latest available modern device drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Qualcomm®	Snapdragon™	Snapdragon 850[1]	
Qualcomm®	Snapdragon™	Snapdragon 7c	
Qualcomm®	Snapdragon™	Snapdragon 7c Gen 2	
Qualcomm®	Snapdragon™	Snapdragon 7c+ Gen 3	
Qualcomm®	Snapdragon™	Snapdragon 8c	
Qualcomm®	Snapdragon™	Snapdragon 8cx	
Qualcomm®	Snapdragon™	Snapdragon 8cx Gen 2	
Qualcomm®	Snapdragon™	Snapdragon 8cx Gen 3	
Qualcomm®	Qualcomm®	QCM6490	IoT Enterprise Only
Qualcomm®	Qualcomm®	QCS6490	IoT Enterprise Only
Qualcomm®	Qualcomm®	QCM5430	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Qualcomm®	Qualcomm®	QCS5430	IoT Enterprise Only
Qualcomm®	Snapdragon™	Microsoft SQ1	
Qualcomm®	Snapdragon™	Microsoft SQ2	
Qualcomm®	Snapdragon™	Microsoft SQ3	

[1] Devices with these CPUs may not be manufactured with Windows 11 pre-installed and may only be upgraded to Windows 11 by a customer.

Windows 11 supported AMD processors

Article • 07/25/2023

The processors listed represent the processor models which meet the minimum floor for the supported processor generations and up through the latest processors at the time of publication. These processors meet the design principles around security, reliability, and the minimum system requirements for Windows 11. Subsequently released and future generations of processors which meet the same principles will be considered as supported, even if not explicitly listed. It is expected that the processor list will not reflect the most current offerings from processor manufacturers between updates. Updates to the processor lists will occur at each subsequent general availability of Windows.

OEMs may use the following CPUs for new Windows 11 devices. New Windows 11 devices must use modern device drivers which have passed the Windows Hardware Compatibility Program for Windows 11 or the latest available modern device drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

Expand table

Manufacturer	Brand	Model
AMD	AMD	3015e
AMD	AMD	3020e
AMD	Athlon™	3000G
AMD	Athlon™	300GE
AMD	Athlon™	300U
AMD	Athlon™	320GE
AMD	Athlon™	7120e
AMD	Athlon™	7120U
AMD	Athlon™	7220e
AMD	Athlon™	7220U
AMD	Athlon™	Gold 3150C
AMD	Athlon™	Gold 3150G
AMD	Athlon™	Gold 3150GE

Manufacturer	Brand	Model
AMD	Athlon™	Gold 3150U
AMD	Athlon™	Silver 3050C
AMD	Athlon™	Silver 3050e
AMD	Athlon™	Silver 3050GE
AMD	Athlon™	Silver 3050U
AMD	Athlon™	Gold PRO 3125GE
AMD	Athlon™	Gold PRO 3150G
AMD	Athlon™	Gold PRO 3150GE
AMD	Athlon™	Gold PRO 4150GE
AMD	Athlon™	PRO 300GE
AMD	Athlon™	PRO 300U
AMD	Athlon™	PRO 3045B
AMD	EPYC™	7252
AMD	EPYC™	7262
AMD	EPYC™	7272
AMD	EPYC™	7282
AMD	EPYC™	7302
AMD	EPYC™	7313
AMD	EPYC™	7343
AMD	EPYC™	7352
AMD	EPYC™	7402
AMD	EPYC™	7413
AMD	EPYC™	7443
AMD	EPYC™	7452
AMD	EPYC™	7453
AMD	EPYC™	7502

Manufacturer	Brand	Model
AMD	EPYC™	7513
AMD	EPYC™	7532
AMD	EPYC™	7542
AMD	EPYC™	7543
AMD	EPYC™	7552
AMD	EPYC™	7642
AMD	EPYC™	7643
AMD	EPYC™	7662
AMD	EPYC™	7663
AMD	EPYC™	7702
AMD	EPYC™	7713
AMD	EPYC™	7742
AMD	EPYC™	7763
AMD	EPYC™	7232P
AMD	EPYC™	72F3
AMD	EPYC™	7302P
AMD	EPYC™	7313P
AMD	EPYC™	73F3
AMD	EPYC™	7402P
AMD	EPYC™	7443P
AMD	EPYC™	74F3
AMD	EPYC™	7502P
AMD	EPYC™	7543P
AMD	EPYC™	75F3
AMD	EPYC™	7702P
AMD	EPYC™	7713P

Manufacturer	Brand	Model
AMD	EPYC™	7F32
AMD	EPYC™	7F52
AMD	EPYC™	7F72
AMD	EPYC™	7H12
AMD	Ryzen Embedded R2000 Series	R2312
AMD	Ryzen Embedded R2000 Series	R2314
AMD	Ryzen Embedded R2000 Series	R2514
AMD	Ryzen Embedded R2000 Series	R2544
AMD	Ryzen™	Z1
AMD	Ryzen™	Z1 Extreme
AMD	Ryzen™ 3	3100
AMD	Ryzen™ 3	4100
AMD	Ryzen™ 3	2300X
AMD	Ryzen™ 3	3200G with Radeon™ Vega 8 Graphics
AMD	Ryzen™ 3	3200GE
AMD	Ryzen™ 3	3200U
AMD	Ryzen™ 3	3250C
AMD	Ryzen™ 3	3250U
AMD	Ryzen™ 3	3300U
AMD	Ryzen™ 3	3350U
AMD	Ryzen™ 3	4300G
AMD	Ryzen™ 3	4300GE
AMD	Ryzen™ 3	4300U
AMD	Ryzen™ 3	5125C
AMD	Ryzen™ 3	5300G (OEM Only)
AMD	Ryzen™ 3	5300GE

Manufacturer	Brand	Model
AMD	Ryzen™ 3	5300GE (OEM Only)
AMD	Ryzen™ 3	5300U
AMD	Ryzen™ 3	5400U
AMD	Ryzen™ 3	5425C
AMD	Ryzen™ 3	5425U
AMD	Ryzen™ 3	7320e
AMD	Ryzen™ 3	7320U
AMD	Ryzen™ 3	7330U
AMD	Ryzen™ 3	7335U
AMD	Ryzen™ 3	7440U
AMD	Ryzen™ 3	5380U
AMD	Ryzen™ 3 PRO	3200G
AMD	Ryzen™ 3 PRO	3200GE
AMD	Ryzen™ 3 PRO	3300U
AMD	Ryzen™ 3 PRO	4350G
AMD	Ryzen™ 3 PRO	4350GE
AMD	Ryzen™ 3 PRO	4450U
AMD	Ryzen™ 3 PRO	5350G
AMD	Ryzen™ 3 PRO	5350GE
AMD	Ryzen™ 3 PRO	5450U
AMD	Ryzen™ 3 PRO	5475U
AMD	Ryzen™ 3 PRO	7330U
AMD	Ryzen™ 3 PRO	4355G
AMD	Ryzen™ 3 PRO	4355GE
AMD	Ryzen™ 5	2600
AMD	Ryzen™ 5	3600

Manufacturer	Brand	Model
AMD	Ryzen™ 5	4500
AMD	Ryzen™ 5	5500
AMD	Ryzen™ 5	5600
AMD	Ryzen™ 5	7600
AMD	Ryzen™ 5	2500X
AMD	Ryzen™ 5	2600E
AMD	Ryzen™ 5	2600X
AMD	Ryzen™ 5	3350G
AMD	Ryzen™ 5	3350GE
AMD	Ryzen™ 5	3400G with Radeon™ RX Vega 11 Graphics
AMD	Ryzen™ 5	3400GE
AMD	Ryzen™ 5	3450U
AMD	Ryzen™ 5	3500 Processor
AMD	Ryzen™ 5	3500C
AMD	Ryzen™ 5	3500U
AMD	Ryzen™ 5	3500X
AMD	Ryzen™ 5	3550H
AMD	Ryzen™ 5	3580U Microsoft Surface® Edition
AMD	Ryzen™ 5	3600X
AMD	Ryzen™ 5	3600XT
AMD	Ryzen™ 5	4500U
AMD	Ryzen™ 5	4600G
AMD	Ryzen™ 5	4600GE
AMD	Ryzen™ 5	4600H
AMD	Ryzen™ 5	4600HS
AMD	Ryzen™ 5	4600U

Manufacturer	Brand	Model
AMD	Ryzen™ 5	5300G
AMD	Ryzen™ 5	5300GE
AMD	Ryzen™ 5	5425U
AMD	Ryzen™ 5	5500U
AMD	Ryzen™ 5	5560U
AMD	Ryzen™ 5	5600G
AMD	Ryzen™ 5	5600GE
AMD	Ryzen™ 5	5600H
AMD	Ryzen™ 5	5600HS
AMD	Ryzen™ 5	5600U
AMD	Ryzen™ 5	5600X
AMD	Ryzen™ 5	5625C
AMD	Ryzen™ 5	5625U
AMD	Ryzen™ 5	6600H
AMD	Ryzen™ 5	6600HS
AMD	Ryzen™ 5	6600U
AMD	Ryzen™ 5	7520U
AMD	Ryzen™ 5	7530U
AMD	Ryzen™ 5	7535HS
AMD	Ryzen™ 5	7535U
AMD	Ryzen™ 5	7540U
AMD	Ryzen™ 5	7600X
AMD	Ryzen™ 5	7640HS
AMD	Ryzen™ 5	7640S
AMD	Ryzen™ 5	7640U
AMD	Ryzen™ 5	7645HX

Manufacturer	Brand	Model
AMD	Ryzen™ 5	7640H
AMD	Ryzen™ 5 PRO	2600
AMD	Ryzen™ 5 PRO	3600
AMD	Ryzen™ 5 PRO	5645
AMD	Ryzen™ 5 PRO	3350G
AMD	Ryzen™ 5 PRO	3350GE
AMD	Ryzen™ 5 PRO	3400G
AMD	Ryzen™ 5 PRO	3400GE
AMD	Ryzen™ 5 PRO	3500U
AMD	Ryzen™ 5 PRO	4650G
AMD	Ryzen™ 5 PRO	4650GE
AMD	Ryzen™ 5 PRO	4650U
AMD	Ryzen™ 5 PRO	5475U
AMD	Ryzen™ 5 PRO	5650G
AMD	Ryzen™ 5 PRO	5650GE
AMD	Ryzen™ 5 PRO	5650HS
AMD	Ryzen™ 5 PRO	5650HX
AMD	Ryzen™ 5 PRO	5650U
AMD	Ryzen™ 5 PRO	5675U
AMD	Ryzen™ 5 PRO	5750G
AMD	Ryzen™ 5 PRO	5750GE
AMD	Ryzen™ 5 PRO	6650H
AMD	Ryzen™ 5 PRO	6650HS
AMD	Ryzen™ 5 PRO	6650U
AMD	Ryzen™ 5 PRO	7530U
AMD	Ryzen™ 5 PRO	7540U

Manufacturer	Brand	Model
AMD	Ryzen™ 5 PRO	7640U
AMD	Ryzen™ 5 PRO	4655G
AMD	Ryzen™ 5 PRO	4655GE
AMD	Ryzen™ 7	2700
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	5800
AMD	Ryzen™ 7	7700
AMD	Ryzen™ 7	2700E Processor
AMD	Ryzen™ 7	2700X
AMD	Ryzen™ 7	3700C
AMD	Ryzen™ 7	3700U
AMD	Ryzen™ 7	3700X
AMD	Ryzen™ 7	3750H
AMD	Ryzen™ 7	3780U Microsoft Surface® Edition
AMD	Ryzen™ 7	3800X
AMD	Ryzen™ 7	3800XT
AMD	Ryzen™ 7	4700G
AMD	Ryzen™ 7	4700GE
AMD	Ryzen™ 7	4700U
AMD	Ryzen™ 7	4800H
AMD	Ryzen™ 7	4800HS
AMD	Ryzen™ 7	4800U
AMD	Ryzen™ 7	5700G
AMD	Ryzen™ 7	5700GE
AMD	Ryzen™ 7	5700U
AMD	Ryzen™ 7	5700X

Manufacturer	Brand	Model
AMD	Ryzen™ 7	5800H
AMD	Ryzen™ 7	5800HS
AMD	Ryzen™ 7	5800U
AMD	Ryzen™ 7	5800X
AMD	Ryzen™ 7	5800X3D
AMD	Ryzen™ 7	5825C
AMD	Ryzen™ 7	5825U
AMD	Ryzen™ 7	6800H
AMD	Ryzen™ 7	6800HS
AMD	Ryzen™ 7	6800U
AMD	Ryzen™ 7	6810U
AMD	Ryzen™ 7	7700X
AMD	Ryzen™ 7	7730U
AMD	Ryzen™ 7	7735HS
AMD	Ryzen™ 7	7735U
AMD	Ryzen™ 7	7736U
AMD	Ryzen™ 7	7745HX
AMD	Ryzen™ 7	7800X3D
AMD	Ryzen™ 7	7840H
AMD	Ryzen™ 7	7840HS
AMD	Ryzen™ 7	7840S
AMD	Ryzen™ 7	7840U
AMD	Ryzen™ 7 PRO	2700
AMD	Ryzen™ 7 PRO	3700
AMD	Ryzen™ 7 PRO	5845
AMD	Ryzen™ 7 PRO	2700X

Manufacturer	Brand	Model
AMD	Ryzen™ 7 PRO	3700U
AMD	Ryzen™ 7 PRO	4750G
AMD	Ryzen™ 7 PRO	4750GE
AMD	Ryzen™ 7 PRO	4750U
AMD	Ryzen™ 7 PRO	5850HS
AMD	Ryzen™ 7 PRO	5850HX
AMD	Ryzen™ 7 PRO	5850U
AMD	Ryzen™ 7 PRO	5875U
AMD	Ryzen™ 7 PRO	6850H
AMD	Ryzen™ 7 PRO	6850HS
AMD	Ryzen™ 7 PRO	6850U
AMD	Ryzen™ 7 PRO	6860Z
AMD	Ryzen™ 7 PRO	7730U
AMD	Ryzen™ 7 PRO	7840U
AMD	Ryzen™ 9	5900
AMD	Ryzen™ 9	7900
AMD	Ryzen™ 9	3900 Processor
AMD	Ryzen™ 9	3900X
AMD	Ryzen™ 9	3900XT
AMD	Ryzen™ 9	3950X
AMD	Ryzen™ 9	4900H
AMD	Ryzen™ 9	4900HS
AMD	Ryzen™ 9	5900HS
AMD	Ryzen™ 9	5900HX
AMD	Ryzen™ 9	5900X
AMD	Ryzen™ 9	5950X

Manufacturer	Brand	Model
AMD	Ryzen™ 9	5980HS
AMD	Ryzen™ 9	5980HX
AMD	Ryzen™ 9	6900HS
AMD	Ryzen™ 9	6900HX
AMD	Ryzen™ 9	6980HS
AMD	Ryzen™ 9	6980HX
AMD	Ryzen™ 9	7845HX
AMD	Ryzen™ 9	7900X
AMD	Ryzen™ 9	7900X3D
AMD	Ryzen™ 9	7940H
AMD	Ryzen™ 9	7940HS
AMD	Ryzen™ 9	7945HX
AMD	Ryzen™ 9	7950X
AMD	Ryzen™ 9	7950X3D
AMD	Ryzen™ 9 PRO	3900
AMD	Ryzen™ 9 PRO	5945
AMD	Ryzen™ 9 PRO	6950H
AMD	Ryzen™ 9 PRO	6950HS
AMD	Ryzen™ Embedded	V2516
AMD	Ryzen™ Embedded	V2546
AMD	Ryzen™ Embedded	V2718
AMD	Ryzen™ Embedded	V2748
AMD	Ryzen™ Threadripper™ PRO	3945WX
AMD	Ryzen™ Threadripper™ PRO	3955WX
AMD	Ryzen™ Threadripper™ PRO	3975WX
AMD	Ryzen™ Threadripper™ PRO	3995WX

Manufacturer	Brand	Model
AMD	Ryzen™ Threadripper™ PRO	5945WX
AMD	Ryzen™ Threadripper™ PRO	5955WX
AMD	Ryzen™ Threadripper™ PRO	5965WX
AMD	Ryzen™ Threadripper™ PRO	5975WX
AMD	Ryzen™ Threadripper™ PRO	5995WX

Windows 11 supported Intel processors

Article • 03/01/2024

The processors listed represent the processor models which meet the minimum floor for the supported processor generations and up through the latest processors at the time of publication. These processors meet the design principles around security, reliability, and the minimum system requirements for Windows 11. Subsequently released and future generations of processors which meet the same principles will be considered as supported, even if not explicitly listed. It is expected that the processor list will not reflect the most current offerings from processor manufacturers between updates. Updates to the processor lists will occur at each subsequent general availability of Windows.

OEMs may use the following CPUs for new Windows 11 devices. New Windows 11 devices must use modern device drivers which have passed the Windows Hardware Compatibility Program for Windows 11 or the latest available modern device drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

Expand table

Manufacturer	Brand	Model
Intel®	Atom®	x6200FE
Intel®	Atom®	x6211E
Intel®	Atom®	x6212RE
Intel®	Atom®	x6413E
Intel®	Atom®	x6414RE
Intel®	Atom®	x6425E
Intel®	Atom®	x6425RE
Intel®	Atom®	x6427FE
Intel®	Celeron®	6305
Intel®	Celeron®	7300
Intel®	Celeron®	7305
Intel®	Celeron®	3867U
Intel®	Celeron®	4205U

Manufacturer	Brand	Model
Intel®	Celeron®	4305U
Intel®	Celeron®	4305UE
Intel®	Celeron®	5205U
Intel®	Celeron®	5305U
Intel®	Celeron®	6305E
Intel®	Celeron®	6600HE
Intel®	Celeron®	7305E
Intel®	Celeron®	7305L
Intel®	Celeron®	G4900
Intel®	Celeron®	G4900T
Intel®	Celeron®	G4920
Intel®	Celeron®	G4930
Intel®	Celeron®	G4930E
Intel®	Celeron®	G4930T
Intel®	Celeron®	G4932E
Intel®	Celeron®	G4950
Intel®	Celeron®	G5900
Intel®	Celeron®	G5900E
Intel®	Celeron®	G5900T
Intel®	Celeron®	G5900TE
Intel®	Celeron®	G5905
Intel®	Celeron®	G5905T
Intel®	Celeron®	G5920
Intel®	Celeron®	G5925
Intel®	Celeron®	G6900
Intel®	Celeron®	G6900E

Manufacturer	Brand	Model
Intel®	Celeron®	G6900T
Intel®	Celeron®	G6900TE
Intel®	Celeron®	J4005[2]
Intel®	Celeron®	J4025
Intel®	Celeron®	J4105[2]
Intel®	Celeron®	J4115[2]
Intel®	Celeron®	J4125
Intel®	Celeron®	J6412
Intel®	Celeron®	J6413
Intel®	Celeron®	N4000[2]
Intel®	Celeron®	N4020
Intel®	Celeron®	N4100[2]
Intel®	Celeron®	N4120
Intel®	Celeron®	N4500
Intel®	Celeron®	N4505
Intel®	Celeron®	N5100
Intel®	Celeron®	N5105
Intel®	Celeron®	N6210
Intel®	Celeron®	N6211
Intel®	Core™	3-100U
Intel®	Core™	3-100U with IPU
Intel®	Core™	5-120U
Intel®	Core™	7-150U
Intel®	Core™	i3-1000G1
Intel®	Core™	i3-1000G4
Intel®	Core™	i3-1005G1

Manufacturer	Brand	Model
Intel®	Core™	i3-10100
Intel®	Core™	i3-10100E
Intel®	Core™	i3-10100F
Intel®	Core™	i3-10100T
Intel®	Core™	i3-10100TE
Intel®	Core™	i3-10100Y
Intel®	Core™	i3-10105
Intel®	Core™	i3-10105F
Intel®	Core™	i3-10105T
Intel®	Core™	i3-10110U
Intel®	Core™	i3-10110Y
Intel®	Core™	i3-10300
Intel®	Core™	i3-10300T
Intel®	Core™	i3-10305
Intel®	Core™	i3-10305T
Intel®	Core™	i3-10320
Intel®	Core™	i3-10325
Intel®	Core™	i3-11100HE
Intel®	Core™	i3-1110G4
Intel®	Core™	i3-1115G4
Intel®	Core™	i3-1115G4E
Intel®	Core™	i3-1115GRE
Intel®	Core™	i3-1120G4
Intel®	Core™	i3-1125G4
Intel®	Core™	i3-12100
Intel®	Core™	i3-12100E

Manufacturer	Brand	Model
Intel®	Core™	i3-12100F
Intel®	Core™	i3-12100T
Intel®	Core™	i3-12100TE
Intel®	Core™	i3-1210U
Intel®	Core™	i3-1215U
Intel®	Core™	i3-1215UE
Intel®	Core™	i3-1215UL
Intel®	Core™	i3-1220P
Intel®	Core™	i3-1220PE
Intel®	Core™	i3-12300
Intel®	Core™	i3-12300HE
Intel®	Core™	i3-12300HL
Intel®	Core™	i3-12300T
Intel®	Core™	i3-1305U
Intel®	Core™	i3-13100
Intel®	Core™	i3-13100E
Intel®	Core™	i3-13100F
Intel®	Core™	i3-13100T
Intel®	Core™	i3-13100TE
Intel®	Core™	i3-1315U
Intel®	Core™	i3-1315UE
Intel®	Core™	i3-1320PE
Intel®	Core™	i3-13300HE
Intel®	Core™	i3-14100
Intel®	Core™	i3-14100F
Intel®	Core™	i3-14100T

Manufacturer	Brand	Model
Intel®	Core™	i3-8100
Intel®	Core™	i3-8100B
Intel®	Core™	i3-8100H
Intel®	Core™	i3-8100T
Intel®	Core™	i3-8109U
Intel®	Core™	i3-8130U
Intel®	Core™	i3-8140U
Intel®	Core™	i3-8145U
Intel®	Core™	i3-8145UE
Intel®	Core™	i3-8300
Intel®	Core™	i3-8300T
Intel®	Core™	i3-8350K
Intel®	Core™	i3-9100
Intel®	Core™	i3-9100E
Intel®	Core™	i3-9100F
Intel®	Core™	i3-9100HL
Intel®	Core™	i3-9100T
Intel®	Core™	i3-9100TE
Intel®	Core™	i3-9300
Intel®	Core™	i3-9300T
Intel®	Core™	i3-9320
Intel®	Core™	i3-9350K
Intel®	Core™	i3-9350KF
Intel®	Core™	i3-N300
Intel®	Core™	i3-N305
Intel®	Core™	i5-10200H

Manufacturer	Brand	Model
Intel®	Core™	i5-10210U
Intel®	Core™	i5-10210Y
Intel®	Core™	i5-10300H
Intel®	Core™	i5-1030G4
Intel®	Core™	i5-1030G7
Intel®	Core™	i5-10310U
Intel®	Core™	i5-10310Y
Intel®	Core™	i5-1035G1
Intel®	Core™	i5-1035G4
Intel®	Core™	i5-1035G7
Intel®	Core™	i5-1038NG7
Intel®	Core™	i5-10400
Intel®	Core™	i5-10400F
Intel®	Core™	i5-10400H
Intel®	Core™	i5-10400T
Intel®	Core™	i5-10500
Intel®	Core™	i5-10500E
Intel®	Core™	i5-10500H
Intel®	Core™	i5-10500T
Intel®	Core™	i5-10500TE
Intel®	Core™	i5-10505
Intel®	Core™	i5-10600
Intel®	Core™	i5-10600K
Intel®	Core™	i5-10600KF
Intel®	Core™	i5-10600T
Intel®	Core™	i5-11260H

Manufacturer	Brand	Model
Intel®	Core™	i5-11300H
Intel®	Core™	i5-1130G7
Intel®	Core™	i5-11320H
Intel®	Core™	i5-1135G7
Intel®	Core™	i5-11400
Intel®	Core™	i5-11400F
Intel®	Core™	i5-11400H
Intel®	Core™	i5-11400T
Intel®	Core™	i5-1140G7
Intel®	Core™	i5-1145G7
Intel®	Core™	i5-1145G7E
Intel®	Core™	i5-1145GRE
Intel®	Core™	i5-11500
Intel®	Core™	i5-11500H
Intel®	Core™	i5-11500HE
Intel®	Core™	i5-11500T
Intel®	Core™	i5-1155G7
Intel®	Core™	i5-11600
Intel®	Core™	i5-11600K
Intel®	Core™	i5-11600KF
Intel®	Core™	i5-11600T
Intel®	Core™	i5-1230U
Intel®	Core™	i5-1235U
Intel®	Core™	i5-1235UL
Intel®	Core™	i5-12400
Intel®	Core™	i5-12400F

Manufacturer	Brand	Model
Intel®	Core™	i5-12400T
Intel®	Core™	i5-1240P
Intel®	Core™	i5-1240U
Intel®	Core™	i5-12450H
Intel®	Core™	i5-12450HX
Intel®	Core™	i5-1245U
Intel®	Core™	i5-1245UE
Intel®	Core™	i5-1245UL
Intel®	Core™	i5-12500
Intel®	Core™	i5-12500E
Intel®	Core™	i5-12500H
Intel®	Core™	i5-12500HL
Intel®	Core™	i5-12500T
Intel®	Core™	i5-12500TE
Intel®	Core™	i5-1250P
Intel®	Core™	i5-1250PE
Intel®	Core™	i5-12600
Intel®	Core™	i5-12600H
Intel®	Core™	i5-12600HE
Intel®	Core™	i5-12600HL
Intel®	Core™	i5-12600HX
Intel®	Core™	i5-12600K
Intel®	Core™	i5-12600KF
Intel®	Core™	i5-12600T
Intel®	Core™	i5-1334U
Intel®	Core™	i5-1335U

Manufacturer	Brand	Model
Intel®	Core™	i5-1335UE
Intel®	Core™	i5-13400
Intel®	Core™	i5-13400E
Intel®	Core™	i5-13400F
Intel®	Core™	i5-13400T
Intel®	Core™	i5-1340P
Intel®	Core™	i5-1340PE
Intel®	Core™	i5-13420H
Intel®	Core™	i5-13450HX
Intel®	Core™	i5-1345U
Intel®	Core™	i5-1345UE
Intel®	Core™	i5-13490F
Intel®	Core™	i5-13500
Intel®	Core™	i5-13500E
Intel®	Core™	i5-13500H
Intel®	Core™	i5-13500HX
Intel®	Core™	i5-13500T
Intel®	Core™	i5-13500TE
Intel®	Core™	i5-13505H
Intel®	Core™	i5-1350P
Intel®	Core™	i5-1350PE
Intel®	Core™	i5-13600
Intel®	Core™	i5-13600H
Intel®	Core™	i5-13600HE
Intel®	Core™	i5-13600HX
Intel®	Core™	i5-13600K

Manufacturer	Brand	Model
Intel®	Core™	i5-13600KF
Intel®	Core™	i5-13600T
Intel®	Core™	i5-14400
Intel®	Core™	i5-14400F
Intel®	Core™	i5-14400T
Intel®	Core™	i5-14500
Intel®	Core™	i5-14500T
Intel®	Core™	i5-14600
Intel®	Core™	i5-14600K
Intel®	Core™	i5-14600KF
Intel®	Core™	i5-14600T
Intel®	Core™	i5-8200Y
Intel®	Core™	i5-8210Y
Intel®	Core™	i5-8250U
Intel®	Core™	i5-8257U
Intel®	Core™	i5-8259U
Intel®	Core™	i5-8260U
Intel®	Core™	i5-8265U
Intel®	Core™	i5-8269U
Intel®	Core™	i5-8279U
Intel®	Core™	i5-8300H
Intel®	Core™	i5-8305G
Intel®	Core™	i5-8310Y
Intel®	Core™	i5-8350U
Intel®	Core™	i5-8365U
Intel®	Core™	i5-8365UE

Manufacturer	Brand	Model
Intel®	Core™	i5-8400
Intel®	Core™	i5-8400B
Intel®	Core™	i5-8400H
Intel®	Core™	i5-8400T
Intel®	Core™	i5-8500
Intel®	Core™	i5-8500B
Intel®	Core™	i5-8500T
Intel®	Core™	i5-8600
Intel®	Core™	i5-8600K
Intel®	Core™	i5-8600T
Intel®	Core™	i5-9300H
Intel®	Core™	i5-9300HF
Intel®	Core™	i5-9400
Intel®	Core™	i5-9400F
Intel®	Core™	i5-9400H
Intel®	Core™	i5-9400T
Intel®	Core™	i5-9500
Intel®	Core™	i5-9500E
Intel®	Core™	i5-9500F
Intel®	Core™	i5-9500T
Intel®	Core™	i5-9500TE
Intel®	Core™	i5-9600
Intel®	Core™	i5-9600K
Intel®	Core™	i5-9600KF
Intel®	Core™	i5-9600T
Intel®	Core™	i7-10510U

Manufacturer	Brand	Model
Intel®	Core™	i7-10510Y
Intel®	Core™	i7-1060G7
Intel®	Core™	i7-10610U
Intel®	Core™	i7-1065G7
Intel®	Core™	i7-1068G7
Intel®	Core™	i7-1068NG7
Intel®	Core™	i7-10700
Intel®	Core™	i7-10700E
Intel®	Core™	i7-10700F
Intel®	Core™	i7-10700K
Intel®	Core™	i7-10700KF
Intel®	Core™	i7-10700T
Intel®	Core™	i7-10700TE
Intel®	Core™	i7-10710U
Intel®	Core™	i7-10750H
Intel®	Core™	i7-10810U
Intel®	Core™	i7-10850H
Intel®	Core™	i7-10870H
Intel®	Core™	i7-10875H
Intel®	Core™	i7-11370H
Intel®	Core™	i7-11375H
Intel®	Core™	i7-11390H
Intel®	Core™	i7-11600H
Intel®	Core™	i7-1160G7
Intel®	Core™	i7-1165G7
Intel®	Core™	i7-11700

Manufacturer	Brand	Model
Intel®	Core™	i7-11700F
Intel®	Core™	i7-11700K
Intel®	Core™	i7-11700KF
Intel®	Core™	i7-11700T
Intel®	Core™	i7-11800H
Intel®	Core™	i7-1180G7
Intel®	Core™	i7-11850H
Intel®	Core™	i7-11850HE
Intel®	Core™	i7-1185G7
Intel®	Core™	i7-1185G7E
Intel®	Core™	i7-1185GRE
Intel®	Core™	i7-1195G7
Intel®	Core™	i7-1250U
Intel®	Core™	i7-1255U
Intel®	Core™	i7-1255UL
Intel®	Core™	i7-1260P
Intel®	Core™	i7-1260U
Intel®	Core™	i7-12650H
Intel®	Core™	i7-12650HX
Intel®	Core™	i7-1265U
Intel®	Core™	i7-1265UE
Intel®	Core™	i7-1265UL
Intel®	Core™	i7-12700
Intel®	Core™	i7-12700E
Intel®	Core™	i7-12700F
Intel®	Core™	i7-12700H

Manufacturer	Brand	Model
Intel®	Core™	i7-12700HL
Intel®	Core™	i7-12700K
Intel®	Core™	i7-12700KF
Intel®	Core™	i7-12700T
Intel®	Core™	i7-12700TE
Intel®	Core™	i7-1270P
Intel®	Core™	i7-1270PE
Intel®	Core™	i7-12800H
Intel®	Core™	i7-12800HE
Intel®	Core™	i7-12800HL
Intel®	Core™	i7-12800HX
Intel®	Core™	i7-1280P
Intel®	Core™	i7-12850HX
Intel®	Core™	i7-1355U
Intel®	Core™	i7-1360P
Intel®	Core™	i7-13620H
Intel®	Core™	i7-13650HX
Intel®	Core™	i7-1365U
Intel®	Core™	i7-1365UE
Intel®	Core™	i7-13700
Intel®	Core™	i7-13700E
Intel®	Core™	i7-13700F
Intel®	Core™	i7-13700H
Intel®	Core™	i7-13700HX
Intel®	Core™	i7-13700K
Intel®	Core™	i7-13700KF

Manufacturer	Brand	Model
Intel®	Core™	i7-13700T
Intel®	Core™	i7-13700TE
Intel®	Core™	i7-13705H
Intel®	Core™	i7-1370P
Intel®	Core™	i7-1370PE
Intel®	Core™	i7-13790F
Intel®	Core™	i7-13800H
Intel®	Core™	i7-13800HE
Intel®	Core™	i7-13850HX
Intel®	Core™	i7-14700
Intel®	Core™	i7-14700F
Intel®	Core™	i7-14700K
Intel®	Core™	i7-14700KF
Intel®	Core™	i7-14700T
Intel®	Core™	i7-7800X
Intel®	Core™	i7-7820HQ[1]
Intel®	Core™	i7-7820X
Intel®	Core™	i7-8086K
Intel®	Core™	i7-8500Y
Intel®	Core™	i7-8550U
Intel®	Core™	i7-8557U
Intel®	Core™	i7-8559U
Intel®	Core™	i7-8565U
Intel®	Core™	i7-8569U
Intel®	Core™	i7-8650U
Intel®	Core™	i7-8665U

Manufacturer	Brand	Model
Intel®	Core™	i7-8665UE
Intel®	Core™	i7-8700
Intel®	Core™	i7-8700B
Intel®	Core™	i7-8700K
Intel®	Core™	i7-8700T
Intel®	Core™	i7-8705G
Intel®	Core™	i7-8706G
Intel®	Core™	i7-8709G
Intel®	Core™	i7-8750H
Intel®	Core™	i7-8809G
Intel®	Core™	i7-8850H
Intel®	Core™	i7-9700
Intel®	Core™	i7-9700E
Intel®	Core™	i7-9700F
Intel®	Core™	i7-9700K
Intel®	Core™	i7-9700KF
Intel®	Core™	i7-9700T
Intel®	Core™	i7-9700TE
Intel®	Core™	i7-9750H
Intel®	Core™	i7-9750HF
Intel®	Core™	i7-9800X
Intel®	Core™	i7-9850H
Intel®	Core™	i7-9850HE
Intel®	Core™	i7-9850HL
Intel®	Core™	i9-10850K
Intel®	Core™	i9-10885H

Manufacturer	Brand	Model
Intel®	Core™	i9-10900
Intel®	Core™	i9-10900E
Intel®	Core™	i9-10900F
Intel®	Core™	i9-10900K
Intel®	Core™	i9-10900KF
Intel®	Core™	i9-10900T
Intel®	Core™	i9-10900TE
Intel®	Core™	i9-10900X
Intel®	Core™	i9-10920X
Intel®	Core™	i9-10940X
Intel®	Core™	i9-10980HK
Intel®	Core™	i9-10980XE
Intel®	Core™	i9-11900
Intel®	Core™	i9-11900F
Intel®	Core™	i9-11900H
Intel®	Core™	i9-11900K
Intel®	Core™	i9-11900KF
Intel®	Core™	i9-11900T
Intel®	Core™	i9-11950H
Intel®	Core™	i9-11980HK
Intel®	Core™	i9-12900
Intel®	Core™	i9-12900E
Intel®	Core™	i9-12900F
Intel®	Core™	i9-12900H
Intel®	Core™	i9-12900HK
Intel®	Core™	i9-12900HX

Manufacturer	Brand	Model
Intel®	Core™	i9-12900K
Intel®	Core™	i9-12900KF
Intel®	Core™	i9-12900KS
Intel®	Core™	i9-12900T
Intel®	Core™	i9-12900TE
Intel®	Core™	i9-12950HX
Intel®	Core™	i9-13900
Intel®	Core™	i9-13900E
Intel®	Core™	i9-13900F
Intel®	Core™	i9-13900H
Intel®	Core™	i9-13900HK
Intel®	Core™	i9-13900HX
Intel®	Core™	i9-13900K
Intel®	Core™	i9-13900KF
Intel®	Core™	i9-13900KS
Intel®	Core™	i9-13900T
Intel®	Core™	i9-13900TE
Intel®	Core™	i9-13905H
Intel®	Core™	i9-13950HX
Intel®	Core™	i9-13980HX
Intel®	Core™	i9-14900
Intel®	Core™	i9-14900F
Intel®	Core™	i9-14900K
Intel®	Core™	i9-14900KF
Intel®	Core™	i9-14900T
Intel®	Core™	i9-7900X

Manufacturer	Brand	Model
Intel®	Core™	i9-7920X
Intel®	Core™	i9-7940X
Intel®	Core™	i9-7960X
Intel®	Core™	i9-7980XE
Intel®	Core™	i9-8950HK
Intel®	Core™	i9-9820X
Intel®	Core™	i9-9880H
Intel®	Core™	i9-9900
Intel®	Core™	i9-9900K
Intel®	Core™	i9-9900KF
Intel®	Core™	i9-9900KS
Intel®	Core™	i9-9900T
Intel®	Core™	i9-9900X
Intel®	Core™	i9-9920X
Intel®	Core™	i9-9940X
Intel®	Core™	i9-9960X
Intel®	Core™	i9-9980HK
Intel®	Core™	i9-9980XE
Intel®	Core™	m3-8100Y
Intel®	Core™	Ultra 5-125H
Intel®	Core™	Ultra 5-125U
Intel®	Core™	Ultra 5-134U
Intel®	Core™	Ultra 5-135H
Intel®	Core™	Ultra 5-135U
Intel®	Core™	Ultra 7-155H
Intel®	Core™	Ultra 7-155U

Manufacturer	Brand	Model
Intel®	Core™	Ultra 7-164U
Intel®	Core™	Ultra 7-165H
Intel®	Core™	Ultra 7-165U
Intel®	Core™	Ultra 9-185H
Intel®	Pentium®	Gold 4417U
Intel®	Pentium®	Gold 4425Y
Intel®	Pentium®	Gold 5405U
Intel®	Pentium®	Gold 6405U
Intel®	Pentium®	Gold 6500Y
Intel®	Pentium®	Gold 6805
Intel®	Pentium®	Gold 7505
Intel®	Pentium®	Gold 8500
Intel®	Pentium®	Gold 8505
Intel®	Pentium®	Gold G5400
Intel®	Pentium®	Gold G5400T
Intel®	Pentium®	Gold G5420
Intel®	Pentium®	Gold G5420T
Intel®	Pentium®	Gold G5500
Intel®	Pentium®	Gold G5500T
Intel®	Pentium®	Gold G5600
Intel®	Pentium®	Gold G5600E
Intel®	Pentium®	Gold G5600T
Intel®	Pentium®	Gold G5620
Intel®	Pentium®	Gold G6400
Intel®	Pentium®	Gold G6400E
Intel®	Pentium®	Gold G6400T

Manufacturer	Brand	Model
Intel®	Pentium®	Gold G6400TE
Intel®	Pentium®	Gold G6405
Intel®	Pentium®	Gold G6405T
Intel®	Pentium®	Gold G6500
Intel®	Pentium®	Gold G6500T
Intel®	Pentium®	Gold G6505
Intel®	Pentium®	Gold G6505T
Intel®	Pentium®	Gold G6600
Intel®	Pentium®	Gold G6605
Intel®	Pentium®	Gold G7400
Intel®	Pentium®	Gold G7400E
Intel®	Pentium®	Gold G7400T
Intel®	Pentium®	Gold G7400TE
Intel®	Pentium®	J6426
Intel®	Pentium®	N6415
Intel®	Pentium®	Silver J5005[2]
Intel®	Pentium®	Silver J5040
Intel®	Pentium®	Silver N5000[2]
Intel®	Pentium®	Silver N5030
Intel®	Pentium®	Silver N6000
Intel®	Pentium®	Silver N6005
Intel®	Processor	N100
Intel®	Processor	N200
Intel®	Processor	N50
Intel®	Processor	N95
Intel®	Processor	N97

Manufacturer	Brand	Model
Intel®	Processor	U300
Intel®	Processor	U300E
Intel®	Xeon®	Bronze 3104
Intel®	Xeon®	Bronze 3106
Intel®	Xeon®	Bronze 3204
Intel®	Xeon®	Bronze 3206R
Intel®	Xeon®	D-1702
Intel®	Xeon®	D-1712TR
Intel®	Xeon®	D-1713NT
Intel®	Xeon®	D-1713NTE
Intel®	Xeon®	D-1714
Intel®	Xeon®	D-1715TER
Intel®	Xeon®	D-1718T
Intel®	Xeon®	D-1722NE
Intel®	Xeon®	D-1726
Intel®	Xeon®	D-1732TE
Intel®	Xeon®	D-1733NT
Intel®	Xeon®	D-1735TR
Intel®	Xeon®	D-1736
Intel®	Xeon®	D-1736NT
Intel®	Xeon®	D-1739
Intel®	Xeon®	D-1746TER
Intel®	Xeon®	D-1747NTE
Intel®	Xeon®	D-1748TE
Intel®	Xeon®	D-1749NT
Intel®	Xeon®	D-2712T

Manufacturer	Brand	Model
Intel®	Xeon®	D-2733NT
Intel®	Xeon®	D-2738
Intel®	Xeon®	D-2752NTE
Intel®	Xeon®	D-2752TER
Intel®	Xeon®	D-2753NT
Intel®	Xeon®	D-2766NT
Intel®	Xeon®	D-2775TE
Intel®	Xeon®	D-2776NT
Intel®	Xeon®	D-2779
Intel®	Xeon®	D-2786NTE
Intel®	Xeon®	D-2795NT
Intel®	Xeon®	D-2796NT
Intel®	Xeon®	D-2796TE
Intel®	Xeon®	D-2798NT
Intel®	Xeon®	D-2799
Intel®	Xeon®	E-2104G
Intel®	Xeon®	E-2124
Intel®	Xeon®	E-2124G
Intel®	Xeon®	E-2126G
Intel®	Xeon®	E-2134
Intel®	Xeon®	E-2136
Intel®	Xeon®	E-2144G
Intel®	Xeon®	E-2146G
Intel®	Xeon®	E-2174G
Intel®	Xeon®	E-2176G
Intel®	Xeon®	E-2176M

Manufacturer	Brand	Model
Intel®	Xeon®	E-2186G
Intel®	Xeon®	E-2186M
Intel®	Xeon®	E-2224
Intel®	Xeon®	E-2224G
Intel®	Xeon®	E-2226G
Intel®	Xeon®	E-2226GE
Intel®	Xeon®	E-2234
Intel®	Xeon®	E-2236
Intel®	Xeon®	E-2244G
Intel®	Xeon®	E-2246G
Intel®	Xeon®	E-2254ME
Intel®	Xeon®	E-2254ML
Intel®	Xeon®	E-2274G
Intel®	Xeon®	E-2276G
Intel®	Xeon®	E-2276M
Intel®	Xeon®	E-2276ME
Intel®	Xeon®	E-2276ML
Intel®	Xeon®	E-2278G
Intel®	Xeon®	E-2278GE
Intel®	Xeon®	E-2278GEL
Intel®	Xeon®	E-2286G
Intel®	Xeon®	E-2286M
Intel®	Xeon®	E-2288G
Intel®	Xeon®	Gold 5115
Intel®	Xeon®	Gold 5118
Intel®	Xeon®	Gold 5119T

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5120
Intel®	Xeon®	Gold 5120T
Intel®	Xeon®	Gold 5122
Intel®	Xeon®	Gold 5215
Intel®	Xeon®	Gold 5215L
Intel®	Xeon®	Gold 5215M
Intel®	Xeon®	Gold 5217
Intel®	Xeon®	Gold 5218
Intel®	Xeon®	Gold 5218B
Intel®	Xeon®	Gold 5218N
Intel®	Xeon®	Gold 5218R
Intel®	Xeon®	Gold 5218T
Intel®	Xeon®	Gold 5220
Intel®	Xeon®	Gold 5220R
Intel®	Xeon®	Gold 5220S
Intel®	Xeon®	Gold 5220T
Intel®	Xeon®	Gold 5222
Intel®	Xeon®	Gold 5315Y
Intel®	Xeon®	Gold 5317
Intel®	Xeon®	Gold 5318N
Intel®	Xeon®	Gold 5318S
Intel®	Xeon®	Gold 5318Y
Intel®	Xeon®	Gold 5320
Intel®	Xeon®	Gold 5320T
Intel®	Xeon®	Gold 5415+
Intel®	Xeon®	Gold 5416S

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 5418Y
Intel®	Xeon®	Gold 5420+
Intel®	Xeon®	Gold 6126
Intel®	Xeon®	Gold 6126F
Intel®	Xeon®	Gold 6126T
Intel®	Xeon®	Gold 6128
Intel®	Xeon®	Gold 6130
Intel®	Xeon®	Gold 6130F
Intel®	Xeon®	Gold 6130T
Intel®	Xeon®	Gold 6132
Intel®	Xeon®	Gold 6134
Intel®	Xeon®	Gold 6136
Intel®	Xeon®	Gold 6138
Intel®	Xeon®	Gold 6138F
Intel®	Xeon®	Gold 6138P
Intel®	Xeon®	Gold 6138T
Intel®	Xeon®	Gold 6140
Intel®	Xeon®	Gold 6142
Intel®	Xeon®	Gold 6142F
Intel®	Xeon®	Gold 6144
Intel®	Xeon®	Gold 6146
Intel®	Xeon®	Gold 6148
Intel®	Xeon®	Gold 6148F
Intel®	Xeon®	Gold 6150
Intel®	Xeon®	Gold 6152
Intel®	Xeon®	Gold 6154

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6208U
Intel®	Xeon®	Gold 6209U
Intel®	Xeon®	Gold 6210U
Intel®	Xeon®	Gold 6212U
Intel®	Xeon®	Gold 6222V
Intel®	Xeon®	Gold 6226
Intel®	Xeon®	Gold 6226R
Intel®	Xeon®	Gold 6230
Intel®	Xeon®	Gold 6230N
Intel®	Xeon®	Gold 6230R
Intel®	Xeon®	Gold 6230T
Intel®	Xeon®	Gold 6234
Intel®	Xeon®	Gold 6238
Intel®	Xeon®	Gold 6238L
Intel®	Xeon®	Gold 6238M
Intel®	Xeon®	Gold 6238R
Intel®	Xeon®	Gold 6238T
Intel®	Xeon®	Gold 6240
Intel®	Xeon®	Gold 6240L
Intel®	Xeon®	Gold 6240M
Intel®	Xeon®	Gold 6240R
Intel®	Xeon®	Gold 6240Y
Intel®	Xeon®	Gold 6242
Intel®	Xeon®	Gold 6242R
Intel®	Xeon®	Gold 6244
Intel®	Xeon®	Gold 6246

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6246R
Intel®	Xeon®	Gold 6248
Intel®	Xeon®	Gold 6248R
Intel®	Xeon®	Gold 6250
Intel®	Xeon®	Gold 6250L
Intel®	Xeon®	Gold 6252
Intel®	Xeon®	Gold 6252N
Intel®	Xeon®	Gold 6254
Intel®	Xeon®	Gold 6256
Intel®	Xeon®	Gold 6258R
Intel®	Xeon®	Gold 6262V
Intel®	Xeon®	Gold 6312U
Intel®	Xeon®	Gold 6314U
Intel®	Xeon®	Gold 6326
Intel®	Xeon®	Gold 6330
Intel®	Xeon®	Gold 6330N
Intel®	Xeon®	Gold 6334
Intel®	Xeon®	Gold 6336Y
Intel®	Xeon®	Gold 6338
Intel®	Xeon®	Gold 6338N
Intel®	Xeon®	Gold 6338T
Intel®	Xeon®	Gold 6342
Intel®	Xeon®	Gold 6346
Intel®	Xeon®	Gold 6348
Intel®	Xeon®	Gold 6354
Intel®	Xeon®	Gold 6416H

Manufacturer	Brand	Model
Intel®	Xeon®	Gold 6418H
Intel®	Xeon®	Gold 6430
Intel®	Xeon®	Gold 6438Y+
Intel®	Xeon®	Gold 6442Y
Intel®	Xeon®	Platinum 8153
Intel®	Xeon®	Platinum 8156
Intel®	Xeon®	Platinum 8158
Intel®	Xeon®	Platinum 8160
Intel®	Xeon®	Platinum 8160F
Intel®	Xeon®	Platinum 8160T
Intel®	Xeon®	Platinum 8164
Intel®	Xeon®	Platinum 8168
Intel®	Xeon®	Platinum 8170
Intel®	Xeon®	Platinum 8171M
Intel®	Xeon®	Platinum 8176
Intel®	Xeon®	Platinum 8176F
Intel®	Xeon®	Platinum 8180
Intel®	Xeon®	Platinum 8253
Intel®	Xeon®	Platinum 8256
Intel®	Xeon®	Platinum 8260
Intel®	Xeon®	Platinum 8260L
Intel®	Xeon®	Platinum 8260M
Intel®	Xeon®	Platinum 8260Y
Intel®	Xeon®	Platinum 8268
Intel®	Xeon®	Platinum 8270
Intel®	Xeon®	Platinum 8272CL

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 8276
Intel®	Xeon®	Platinum 8276L
Intel®	Xeon®	Platinum 8276M
Intel®	Xeon®	Platinum 8280
Intel®	Xeon®	Platinum 8280L
Intel®	Xeon®	Platinum 8280M
Intel®	Xeon®	Platinum 8351N
Intel®	Xeon®	Platinum 8352M
Intel®	Xeon®	Platinum 8352S
Intel®	Xeon®	Platinum 8352V
Intel®	Xeon®	Platinum 8352Y
Intel®	Xeon®	Platinum 8358
Intel®	Xeon®	Platinum 8358P
Intel®	Xeon®	Platinum 8360Y
Intel®	Xeon®	Platinum 8362
Intel®	Xeon®	Platinum 8368
Intel®	Xeon®	Platinum 8368Q
Intel®	Xeon®	Platinum 8380
Intel®	Xeon®	Platinum 8452Y
Intel®	Xeon®	Platinum 8460Y+
Intel®	Xeon®	Platinum 8468
Intel®	Xeon®	Platinum 8470
Intel®	Xeon®	Platinum 8480+
Intel®	Xeon®	Platinum 8490H
Intel®	Xeon®	Platinum 9221
Intel®	Xeon®	Platinum 9222

Manufacturer	Brand	Model
Intel®	Xeon®	Platinum 9242
Intel®	Xeon®	Platinum 9282
Intel®	Xeon®	Silver 4108
Intel®	Xeon®	Silver 4109T
Intel®	Xeon®	Silver 4110
Intel®	Xeon®	Silver 4112
Intel®	Xeon®	Silver 4114
Intel®	Xeon®	Silver 4114T
Intel®	Xeon®	Silver 4116
Intel®	Xeon®	Silver 4116T
Intel®	Xeon®	Silver 4208
Intel®	Xeon®	Silver 4209T
Intel®	Xeon®	Silver 4210
Intel®	Xeon®	Silver 4210R
Intel®	Xeon®	Silver 4210T
Intel®	Xeon®	Silver 4214
Intel®	Xeon®	Silver 4214R
Intel®	Xeon®	Silver 4214Y
Intel®	Xeon®	Silver 4215
Intel®	Xeon®	Silver 4215R
Intel®	Xeon®	Silver 4216
Intel®	Xeon®	Silver 4309Y
Intel®	Xeon®	Silver 4310
Intel®	Xeon®	Silver 4310T
Intel®	Xeon®	Silver 4314
Intel®	Xeon®	Silver 4316

Manufacturer	Brand	Model
Intel®	Xeon®	Silver 4410T
Intel®	Xeon®	Silver 4410Y
Intel®	Xeon®	Silver 4416+
Intel®	Xeon®	W-10855M
Intel®	Xeon®	W-10885M
Intel®	Xeon®	W-11055M
Intel®	Xeon®	W-11155MLE
Intel®	Xeon®	W-11155MRE
Intel®	Xeon®	W-11555MLE
Intel®	Xeon®	W-11555MRE
Intel®	Xeon®	W-11855M
Intel®	Xeon®	W-11865MLE
Intel®	Xeon®	W-11865MRE
Intel®	Xeon®	W-11955M
Intel®	Xeon®	W-1250
Intel®	Xeon®	W-1250E
Intel®	Xeon®	W-1250P
Intel®	Xeon®	W-1250TE
Intel®	Xeon®	W-1270
Intel®	Xeon®	W-1270E
Intel®	Xeon®	W-1270P
Intel®	Xeon®	W-1270TE
Intel®	Xeon®	W-1290
Intel®	Xeon®	W-1290E
Intel®	Xeon®	W-1290P
Intel®	Xeon®	W-1290T

Manufacturer	Brand	Model
Intel®	Xeon®	W-1290TE
Intel®	Xeon®	W-1350
Intel®	Xeon®	W-1350P
Intel®	Xeon®	W-1370
Intel®	Xeon®	W-1370P
Intel®	Xeon®	W-1390
Intel®	Xeon®	W-1390P
Intel®	Xeon®	W-1390T
Intel®	Xeon®	W-2102
Intel®	Xeon®	W-2104
Intel®	Xeon®	W-2123
Intel®	Xeon®	W-2125
Intel®	Xeon®	W-2133
Intel®	Xeon®	W-2135
Intel®	Xeon®	W-2145
Intel®	Xeon®	W-2155
Intel®	Xeon®	W-2175
Intel®	Xeon®	W-2195
Intel®	Xeon®	W-2223
Intel®	Xeon®	W-2225
Intel®	Xeon®	W-2235
Intel®	Xeon®	W-2245
Intel®	Xeon®	W-2255
Intel®	Xeon®	W-2265
Intel®	Xeon®	W-2275
Intel®	Xeon®	W-2295

Manufacturer	Brand	Model
Intel®	Xeon®	W-3175X
Intel®	Xeon®	W-3223
Intel®	Xeon®	W-3225
Intel®	Xeon®	W-3235
Intel®	Xeon®	W3-2423
Intel®	Xeon®	W3-2425
Intel®	Xeon®	W3-2435
Intel®	Xeon®	W-3245
Intel®	Xeon®	W-3245M
Intel®	Xeon®	W-3265
Intel®	Xeon®	W-3265M
Intel®	Xeon®	W-3275
Intel®	Xeon®	W-3275M
Intel®	Xeon®	W-3323
Intel®	Xeon®	W-3335
Intel®	Xeon®	W-3345
Intel®	Xeon®	W-3365
Intel®	Xeon®	W-3375
Intel®	Xeon®	W5-2445
Intel®	Xeon®	W5-2455X
Intel®	Xeon®	W5-2465X
Intel®	Xeon®	W5-3423
Intel®	Xeon®	W5-3425
Intel®	Xeon®	W5-3433
Intel®	Xeon®	W5-3435X
Intel®	Xeon®	W7-2475X

Manufacturer	Brand	Model
Intel®	Xeon®	W7-2495X
Intel®	Xeon®	W7-3445
Intel®	Xeon®	W7-3455
Intel®	Xeon®	W7-3465X
Intel®	Xeon®	W9-3475X
Intel®	Xeon®	W9-3495X

[1] Only select devices that shipped with modern drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

[2] Devices with these CPUs may not be manufactured with Windows 11 pre-installed and may only be upgraded to Windows 11 by a customer.

Windows 11 supported Qualcomm processors

Article • 09/28/2023

The processors listed represent the processor models which meet the minimum floor for the supported processor generations and up through the latest processors at the time of publication. These processors meet the design principles around security, reliability, and the minimum system requirements for Windows 11. Subsequently released and future generations of processors which meet the same principles will be considered as supported, even if not explicitly listed. It is expected that the processor list will not reflect the most current offerings from processor manufacturers between updates. Updates to the processor lists will occur at each subsequent general availability of Windows.

OEMs may use the following CPUs for new Windows 11 devices. New Windows 11 devices must use modern device drivers which have passed the Windows Hardware Compatibility Program for Windows 11 or the latest available modern device drivers based on Declarative, Componentized, Hardware Support Apps (DCH) design principles.

Expand table

Manufacturer	Brand	Model	IoT Applicability
Qualcomm®	Snapdragon™	Snapdragon 850[1]	
Qualcomm®	Snapdragon™	Snapdragon 7c	
Qualcomm®	Snapdragon™	Snapdragon 7c Gen 2	
Qualcomm®	Snapdragon™	Snapdragon 7c+ Gen 3	
Qualcomm®	Snapdragon™	Snapdragon 8c	
Qualcomm®	Snapdragon™	Snapdragon 8cx	
Qualcomm®	Snapdragon™	Snapdragon 8cx Gen 2	
Qualcomm®	Snapdragon™	Snapdragon 8cx Gen 3	
Qualcomm®	Qualcomm®	QCM6490	IoT Enterprise Only
Qualcomm®	Qualcomm®	QCS6490	IoT Enterprise Only
Qualcomm®	Qualcomm®	QCM5430	IoT Enterprise Only

Manufacturer	Brand	Model	IoT Applicability
Qualcomm®	Qualcomm®	QCS5430	IoT Enterprise Only
Qualcomm®	Snapdragon™	Microsoft SQ1	
Qualcomm®	Snapdragon™	Microsoft SQ2	
Qualcomm®	Snapdragon™	Microsoft SQ3	

[1] Devices with these CPUs may not be manufactured with Windows 11 pre-installed and may only be upgraded to Windows 11 by a customer.

Form factors

Article • 01/18/2019

Form factor describes the size of a device and its physical (hardware) components. Windows 10 supports a wide range of devices from small embedded devices, traditional notebooks to large servers. Depending on the device you choose, you have a wide range of options and requirements. This section defines common form factors and provides example hardware configuration for each type of form factor.

In this section

Topic	Description
2-in-1	A 2-in-1 form factor is an evolution of the traditional notebook, also know as a convertible notebook that comes in two styles: detachable and convertible. Its key feature is the ability to convert from a traditional notebook to a tablet mode.
All-in-one	The All-in-one (AIO) form factor is an evolution of the traditional desktop with the benefit of taking up less space. It combines all the system components and display into one chassis, similar to a tablet.
Desktop PC	A desktop PC form factor traditionally comes in an upright tower or small desktop chassis.
Notebook	A notebook is a portable clamshell device with an attached keyboard.
Tablet	Tablets are a versatile form factor that overlap the smartphone and notebook industry. A tablet device combines a touch screen, rechargeable power source, and other components into a single chassis with an optional attachable keyboard. Listed below are a few recommended hardware configurations.
Servers	The server form factor comes in a wide range of styles including pedestal, rackmount and blade.
Smartphone	Smartphones come in a wide range of sizes from low cost phones, small tablets (phablets) and industrial handheld devices.

Topic	Description
Embedded devices	Windows 10 IoT Core (IoT Core) can be used on low cost, small footprint devices that support Universal Apps.

2-in-1

Article • 08/24/2021

A 2-in-1 form factor is an evolution of the traditional notebook, also know as a convertible notebook that comes in two styles: detachable and convertible. Its key feature is the ability to convert from a traditional notebook to a tablet mode. Convertibles are typically more like a notebook that transforms into a tablet. Detachables are typically more like a tablet with a high quality keyboard base or attachment. Windows 10 for desktop editions (Home, Pro, Enterprise, and Education) introduces a new feature called Continuum that takes advantage of a 2-in-1s unique design. The feature will adjust the user experience based on what mode the device is in. The following table describes some recommended hardware configurations.

Expand table

Type	Entry 2-in-1 Detachable	Mainstream 2-in-1	Premium 2-in-1
Key Features	Continuum Windows Hello Wired and Wireless Docking and Casting (USB-C; Miracast) Inking/Pen Support	Continuum Windows Hello Wired and Wireless Docking and Casting (USB-C; Miracast) Inking/Pen Support	Continuum Windows Hello Wired and Wireless Docking and Casting (USB-C; Miracast) Inking/Pen Support Long battery life (12+ hours) Precision Touchpad
Operating System	Windows 10 for desktop editions		
Recommended components			
CPU	Mobile to mid-tier x86	Mid-range x86	Premium x86

RAM/Storage	2 GB+ / 32 GB+ or 320 GB+ HDD	4 GB+ / 32 GB+ with SD slot	4–16 GB / 64 GB–1 TB SSD
Display	10.1”–13.3” HD+ Touch	10.1”–12.5” FHD+	11.6”–14” FHD-4K / Touch
Dimensions	< 10mm & < .6kg (Tablet)	< 18mm & < 1.36kgs w/ keyboard	< 16mm & < 1.36kg (combined w/ keyboard)
Battery	8+ hours	8+ hours	12+ hours
Connectivity	802.11ac 1 USB 3.0 BT LE HDMI LTE option	802.11ac USB 3.0 BT LE HDMI LTE option NFC	802.11ac 2+ USB 3.*x* BT LE LTE option
Audio/Video	Front & rear cameras Speakers Headphones Full-array mics	Front & rear cameras Speakers Headphones Full-array mics	Stereo Speaker HD Webcam Full-array mics

For more guidance in selecting hardware components, see Hardware component guidelines.

All-in-one

Article • 08/24/2021

The All-in-one (AIO) form factor is an evolution of the traditional desktop with the benefit of taking up less space. It combines all the system components and display into one chassis, similar to a tablet. Listed below are a few recommended hardware configurations.

Type	Portable All-in-one	Premium All-in-one
Key Features	Portable (< 5 pounds, thin) Windows Hello Battery Life (6+ hours) Multi-user, multi-touch apps (Sensors, hinge design for "lay-flat," 10-point touch)	High-DPI (4K) & multi-monitor support Windows Hello Inking/Pen Support Multi-user, multi-touch apps
Operating System	Windows 10 for desktop editions (Home, Pro, Enterprise, and Education)	
Recommended components		
CPU	Mid-tier x86 w/CS	High-end x86
RAM/Storage	2 GB+ / 320 GB+ HDD or 64 GB+ SSD	8 GB+ / 256 GB+ SSD or 1 TB+ HDD
Display</P	17"–21" / HD Touch	21"–30" / FHD or QHD Touch, DirectX 12
Dimensions	< 21 mm & < 2.3 kg	Sleek design
Battery	6+ hours	-
Connectivity	802.11ac USB 3.0	802.11ac 1 USB 3.0

	HDMI BT LE	HDMI BT LE
Audio/Video	HD Webcam	HD Webcam Premium Speakers full-array mics

For more guidance in selecting hardware components, see Hardware component guidelines.

Desktop PC

Article • 08/24/2021

A desktop PC form factor traditionally comes in an upright tower or small desktop chassis. Listed below are a few recommended hardware configurations.

Type	Mainstream desktop	Mini PC desktop
Key Features	Full PC experience (Core performance, legacy ports, optical disk drive)	Full PC experience (Core performance with optimal storage)
Operating System	Windows 10 for desktop editions (Home, Pro, Enterprise, and Education)	
Recommended components		
CPU	Mid-range x86	Mid-range x86
RAM/Storage	4 GB+ / 32 GB+ SSD or 500 GB+ HDD	4 GB / 32+ GB SSD or 320 GB+ HDD
Display	Multi-monitor support	-
Dimensions	-	< 2.5 L compact size
Connectivity	802.11ac USB 3.0 HDMI BT	802.11ac USB 3.0 HDMI BT LE

For more guidance in selecting hardware components, see Hardware component guidelines.

Notebook

Article • 08/24/2021

A notebook is a portable clamshell device with an attached keyboard. The notebook industry continues to grow with a wide range of options from inexpensive notebooks, mainstream notebooks to gaming notebooks. The advances in technology and manufacturing have brought notebooks closer to desktops. Listed below are a few recommended hardware configurations.

Expand table

Type	Entry Notebook	Mainstream Notebook	Premium Ultra-slim Notebook	Workstation Notebook	Gaming Notebook
Key Features	Windows Hello Battery Life 8+ hours Precision Touchpad	Wired and Wireless Docking and Casting (USB-C; Miracast) Inking/Pen Support Precision Touchpad	High-DPI (4K) Long battery life (12+ hours) Wired and Wireless Docking and Casting (USB-C; Miracast) Windows Hello Precision Touchpad	Wired and Wireless Docking and Casting (USB-C; Miracast) Inking/Pen Support Precision Touchpad	Premium A/V experience (DirectX 12; 4K support, display & chipset), premium speakers, chipset support for PlayReady, H/W DRM, H.265 H/W decode/encode) Xbox content, Xbox Controller, Xbox Live Windows Hello Precision Touchpad

Operating System	Windows 10 for desktop editions (Home, Pro, Enterprise, and Education)				
Recommended components					
CPU	Mid-tier x86	Mid-range x86	Premium high-end x86	Premium high-end x86	Premium high-end x86
RAM/Storage	2 GB+ / 32 GB+ eMMC or SSD + SD Card	4 GB+ / 500 GB+ HDD	8 GB / 256 GB SSD or 1 TB HD w/SSD cache	8+ GB / 256 GB+ SSD or 750 GB–1 TB+ HDD + SSD cache	8 GB+ / 1 TB+ HDD + SSD cache
Display	11.6"+ / HD+ non-touch	14.1"–15.6" / FHD+ Touch	11.6"–12.5" display w/touch; FHD-WQHD, zero-bezel	14" / FHD+	14"–19" / FHD – 4K Discrete DirectX 12 GPU with 2 GB–4 GB
Dimensions	< 18 mm & < 1.36 kg	< 19 mm & 1.8 kg	< 15 mm & < .51 kgs	< 20 mm & < 1.5 kg	< 20 mm & < 1.5 kg
Battery	8+ hours	6+ hours	12+ hours	-	-
Connectivity	802.11ac 1 USB 3.0 HDMI BT LE	802.11ac USB 3.0 HDMI BT LE NFC	802.11ac 2+ USB 3.*x* HDMI BT LE LTE	802.11ac 1 USB 3.0 HDMI BT LE	802.11ac 1 USB 3.0 HDMI BT LE LTE
Audio/Video	Stereo speaker Full-array mics HD Webcam	Webcam Speakers Headphones	Stereo speaker Full-array mics HD Webcam	Premium sound HD Webcam Full-array mics	Premium sound HD Webcam Full-array mics

For more guidance in selecting hardware components, see Hardware component guidelines.

Tablet

Article • 08/24/2021

Tablets are a versatile form factor that overlap the smartphone and notebook industry. A tablet device combines a touch screen, rechargeable power source, and other components into a single chassis with an optional attachable keyboard. Listed below are a few recommended hardware configurations.

	7" Tablet	8" Tablet	Large Tablet
Key Features	Low cost Continuum for Phones	Battery Life 8+ hours Wired and Wireless Docking and Casting (USB-C; Miracast)	Ruggedized for commercial and educational use Inking/Pen Support Windows Hello Wired and Wireless Docking and Casting (USB-C; Miracast)
Operating System	Windows 10 Mobile	Windows 10 for desktop editions (Home, Pro, Enterprise, and Education)	
Recommended components			
CPU	Supported entry SoC	Entry x86	Entry x86
RAM/Storage	1-2 GB / 8-32 GB eMMC w/SD card	1 GB+ / 32 GB+ SSD	4 GB+ / 32 GB+ SSD
Display	7” 480×800 or 1280×720 w/touch	8” / HD+	10.1”–12.5” / FHD+
Dimensions	< 9 mm & < .36 kg	< 9 mm & < .6 kg	-

Battery	10+ hours	8+ hours	8+ hours
Connectivity	802.11ac+	802.11ac	802.11ac
	1 micro USB 2.0	USB 3.0	USB 3.0
	mini HDMI	HDMI	HDMI
	BT	BT LE	BT LE
	LTE option	NFC	NFC
		LTE Option	LTE Option
Audio/Video	Front camera	Front camera	Front & rear camera
	Speakers	Speakers	Speakers
	Headphones	Headphones	Headphones
		Full-array mics	Full-array mics

For more guidance in selecting hardware components, see Hardware component guidelines.

Research the Windows Mobile options and partner with a registered Windows phone manufacturer to build a device utilizing your brand. For more information, see Windows 10 Hardware Developer Programs .

Servers

Article • 11/05/2020

The server form factor comes in a wide range of styles including pedestal, rackmount and blade.

Server form factors usually include high performance/capacity components. For example,

- Larger power supply
- Larger fan(s)
- Multiple processor support
- Multiple storage bays

For detailed information on server hardware requirements, see

- Minimum Hardware Requirements Overview
- Windows Hardware Compatibility Program Requirements

Embedded devices

Article • 01/25/2019

Windows 10 IoT Core (IoT Core) can be used on low cost, small footprint devices that support Universal Apps. For more information, visit https://windowsondevices.com .

Device experiences

Article • 10/20/2022

To build devices with a rich user experience, you need to choose the right hardware for usability and performance. This section describes important device experience scenarios and highlights hardware that takes advantage of Windows features.

In this section

Topic	Description
Docking	Provides guidance on wired and wireless docking scenarios.
Tablet mode	Tablet mode is a new, adaptive user experience offered in Windows 10 that optimizes the look and behavior of apps and the Windows shell for the physical form factor and customer's usage preferences. This document describes how to implement tablet mode on 2-in-1 devices and tablets, specifically how to switch in and out of "tablet mode."
Windows Hello	Microsoft Windows Hello, part of Windows 10, gives users a personal, secured experience where the device is authenticated based on their presence. Users can log in with a look or a touch, with no need for a password. In conjunction with Microsoft Passport, biometric authentication uses fingerprints or facial recognition and is more secure, more personal, and more convenient. Windows Hello can even give users a friendly hello and greet them by name.
Modern standby	Modern standby (MS) expands the Windows 8.1 connected standby power model to be more inclusive and allows systems based on rotational media and hybrid media (for example, SSD + HDD or SSHD), or a NIC that doesn't support all of the prior requirements for connected standby to still take advantage of the low power idle model. In modern standby, the PC uses the S0 low power idle model. Modern standby has the flexibility to configure the default behavior to limit network activity while in the low power state.
Thermal management in Windows	Provides information about how thermal management in Windows plays a key role in delivering PCs that have good performance and are safe to use even when they are running a high-energy workload.
Security considerations for OEMs	Provides guidance on platform integrity and data protection features including secure boot and BitLocker.
Versatile input platform	Provides guidance on touchpad device experiences.

Topic	Description
Wireless projection	Wireless projection allows a user to mirror their screen or application to an array of receivers such as media devices or TVs. This creates the ability to project content from a PC to a larger screen for the purposes of sharing or collaboration.
Adaptive brightness	Adaptive brightness is the screen brightness set automatically by the system in response to the ambient light sensor reading. Adaptive brightness provides users a more responsive display experience where brightness adapts automatically to the user's environment.

Docking and multi-monitor Experiences

Article • 06/28/2021

Overview

The docking and multiple monitor experience is improved in Windows 11 to make the monitor attach and removal process smoother and more reliable by adding clean animations and also preserving the user's window arrangements on a per-monitor layout basis.

Today's ecosystem of cables, docks, and different port requirements can make it difficult for a customer to understand which pieces are compatible with the device that they purchase. If a customer uses a connector that is below standard, they may see a lower resolution on their monitor (the max resolution of the monitor may not be supported), a lower refresh rate than advertised, or screen flickering and other visual artifacts.

Recommended Practices

When messaging supported features and scenarios, drive as much clarity as possible into messaging, especially for technical capabilities like 4K and higher refresh rates. Many different pieces of hardware advertise 4K support, but 4K can only be achieved in a limited set of scenarios. For example, if a docking station has multiple devices connected it might not support being connected to two 4K monitors and ethernet and USB all at the same time. It can be confusing for people to navigate the guaranteed scenarios versus the maximum/optimal scenarios, so better clarity will help.

Bundle and Port Recommendations

If you promote bundles that include laptops, docking station/connector cables, and external monitors, then here are a few recommendations for port requirements and connection types:

The 'Recommended Practice' column for Port Standards is intended to promote a great experience, so while it might be possible to have a good docking experience with ports at lower standards, it might not be possible in all scenarios, for example, in the scenario described above where a person has plugged in multiple things to their docking station.

Scenario: Premium setup with a 4K monitor and a laptop

This is one of our star scenarios that we want to highlight the amazing capabilities of the hardware and software. We especially want to ensure that the experience is smooth, efficient, and effortless for customers.

Expand table

	Recommended Practice	Good Practice	Not-recommended Practice
Port Standards	DisplayPort 1.2+, HDMI 2.1+, Thunderbolt 3+, USB4	--	Port versions below recommended practice might have mixed results depending on what else is connected to the docking station.
Connection Type	Direct cable connections are best.	Docks will be possible, but might be risky with HDR.	Converter cables should be avoided if possible.

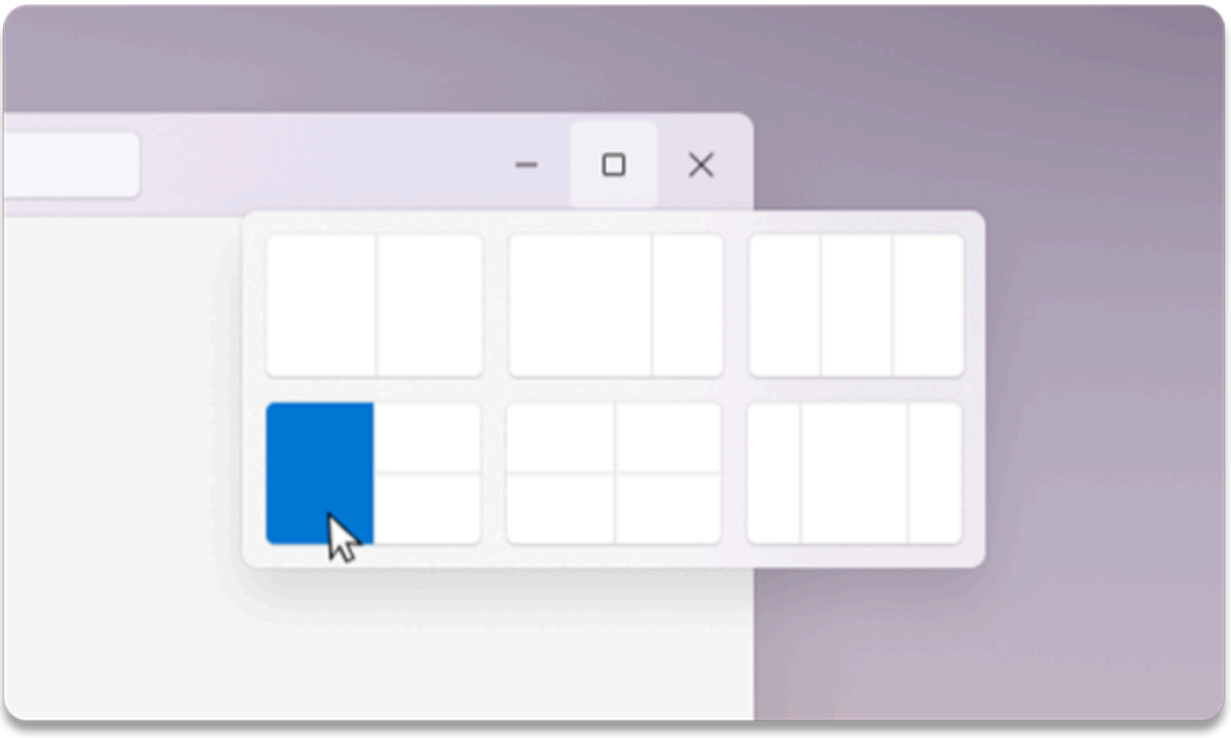
Scenario: Premium set up with two 4K monitors and a laptop

Expand table

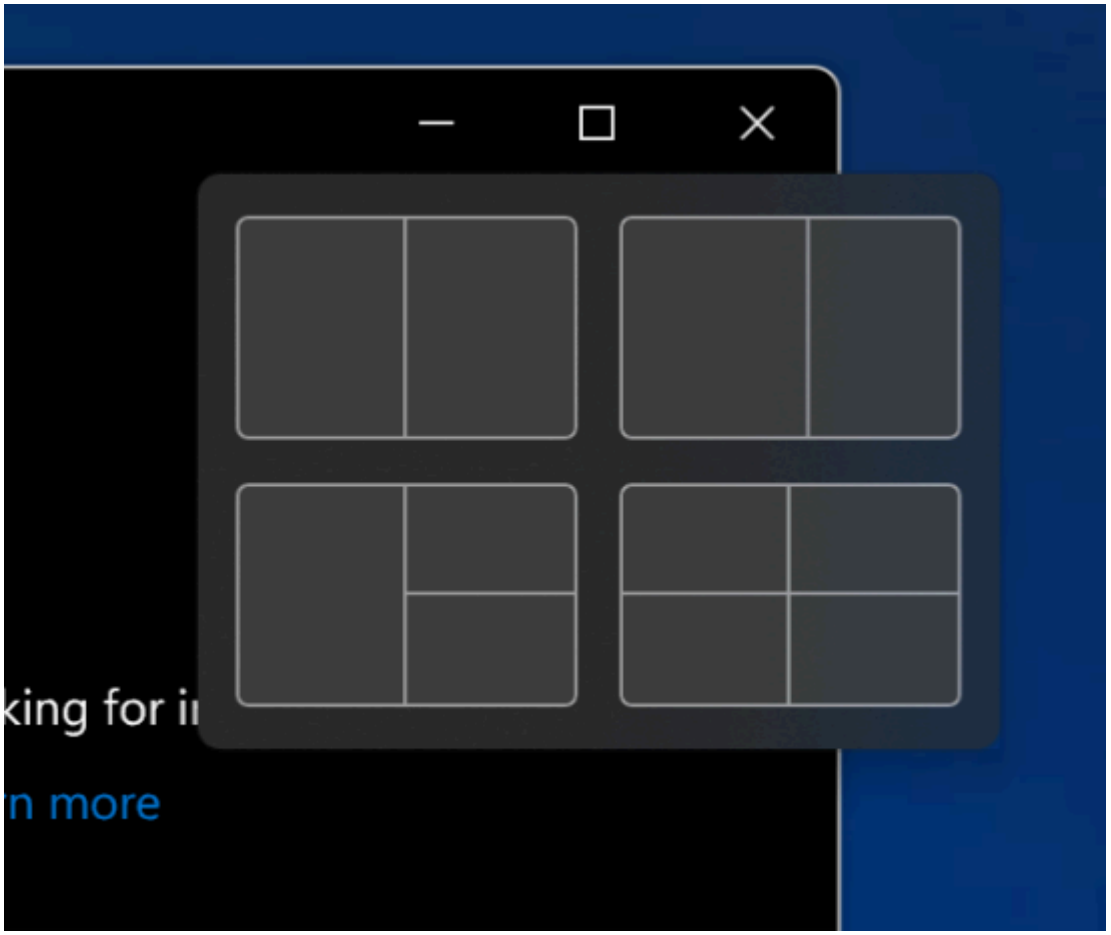
	Best Practice	Good Practice	Not-recommended Practice
Port Requirements	DisplayPort 2.0+, HDMI 2.1+, Thunderbolt 3+, USB4	--	Port versions below recommendations might have mixed results
Connection Type	Direct cable connections	Docks will be risky but are possible	Converter cables should be avoided if possible.

Snap Layouts and Controls

Windows now has an easier way to snap your app windows. The new layouts are visible when you hover the mouse over a window's maximize button. When you click a zone to snap the window into place, you will be guided to snap windows to the rest of the zones in your chosen layout with guided snap assist. On larger displays, you'll also see new snap layouts with 3 columns of zones. In addition to the mouse hover on the maximize button, you can also invoke the snap flyout by pressing Win + Z. The following picture shows the snap layout choices available on displays that are larger than 24 inches and at 1920 effective pixels.



The following picture shows the snap layout choices that are available on displays smaller than 24 inches and at 1920 effective pixels.



Snap Groups

Snap groups provide greater ease in switching back to snapped windows. To use this feature, snap together at least 2 windows on your screen. Hover over one of your open apps in the Taskbar to find the snap group and click to switch back.

Docking

Article • 11/05/2020

To enable fast, seamless connections between desktops and mobile devices to external equipment, Windows 10 supports single cable docking and wireless docking. Interoperability is the key goal of the docking recommendations included here. These recommendations comprise both system and dock recommendations to ensure that a user can confidently dock their desktop system devices and their mobile system devices.

A dock is a device that is intended to enhance the productivity of a system. At a minimum, a dock has the following capabilities:

1. Contains a display-out option or a built-in monitor
2. Allows the user to connect to external peripheral devices

The dock manufacturer is responsible for properly advertising and marketing the type of display and peripheral devices that are supported by the dock.

Wired docking

A wired docking configuration uses a single connection to a dock. The connection is either through a cable or a device to an adjacent dock attachment. The dock provides ports for input devices, such as mice and keyboards, and for output devices like displays and printers. Better docking equipment provides a power connection, for both use and charging of the device, and a wired network connection.

Wired docking features to consider:

- Use of a single cable from the device to the dock
- Support for phone, tablet, and laptops
- Wired or wireless mouse and keyboard input
- Multiple connections on the dock
- Monitor outputs using HDMI, DisplayPort, or both
- A power connection to the device
- An Ethernet port

Overall recommendations

The device-to-dock connection should be USB Type-C. The introduction of the USB Type-C connector allows us to truly reach a single-cable solution that supports docks, displays, data devices, and power for the majority of portable systems (phones, tablets,

and laptops). We expect the largest changes in the wired docking space to be around USB Type-C alternate modes, with minor changes coming with updates of graphics standards.

USB Type-C introduced the concept of alternate modes (protocols other than USB over a symmetrical and reversible Type-C connector and cable), but because this technology is still "young," we expect to see other alternate modes defined in the future. As more alternate modes are added, we will continue to reevaluate our USB Type-C recommendations.

Currently, we are recommending that DisplayPort over Type-C be used as the alternate mode for docks and displays. Although we do allow docks and displays to use other USB Type-C alternate modes, these devices must be able to fall back to the DisplayPort over Type-C alternate mode to ensure interoperability. For example, Thunderbolt 3 docks and displays must still work when the dock or display is connected to a system that only supports DisplayPort over Type-C alternate mode.

Additionally, Windows 10, version 1607 supports Indirect Display over USB, so any dock that supports Indirect Display over USB must also have a fallback to DisplayPort. Indirect Display is a simple user-mode driver model to support monitors that are not connected to traditional graphics processing unit (GPU) display outputs. An example would be a USB dongle connected to a PC that has a regular Video Graphics Array (VGA), Digital Visual Interface (DVI), High-Definition Multimedia Interface (HDMI), or DisplayPort monitor attached to it. The Indirect Display interface works over all USB ports, including Type A and Type-C (with and without alternate mode). For more information about Indirect Display, see Indirect Display Driver Model Overview.

As HDMI and DisplayPort protocol specifications mature, our recommendations will change to take advantage of new features.

Wired docking system recommendations

All systems should meet at least the minimum hardware requirements and the Windows Hardware Compatibility Specifications. In addition to these requirements, we recommend that systems work with docks built to the wired docking recommendations. This section covers what additionally is needed on the system side.

Expand table

System connection to the dock	The system should include at least one USB Type-C port that supports USB Host or USB Dual Role/USB On-The-Go (OTG). All USB Type-C ports should include support for the following: • USB Data, with USB 3.1 Gen 2 being optimal

	• USB Power Delivery (PD) power sink to allow systems to charge using USB-C from dock, so that a user only needs to connect one wire to their system to be productive for several hours • DisplayPort Alternate Mode, pin assignment C, D, E, and F, with support for High Bit Rate (HBR) and High Bit Rate 2 (HBR2) signaling on those pin assignments, and the ability to source DisplayPort on at least two DisplayPort lanes for all supported pin assignments For systems with multiple USB Type-C ports, we recommend that all ports support the specifications above. If the specifications are not supported by all of the included USB Type-C ports, we recommend that these ports be visually differentiable to the user as defined in the USB-IF Trademark License Agreement under the section "Logo Guidelines for USB Type-C Products and Cables that Support Alternate Modes." If you plan on including an alternate mode other than DisplayPort over USB Type-C alternate mode, review the fallback requirements in the Overall recommendations section above. For more information about implementing USB Type-C, see Developing Windows drivers for USB Type-C connectors on MSDN.
RAM	The system should have at least 2 GB of RAM.
Flash storage	The system should have at least 16 GB of nonremovable flash memory.
Audio/video latency	Audio or video latency should not exceed 80 milliseconds.

Wired dock (device) recommendations

The end goal of these docking recommendations is to maximize compatibility and interoperability. For docks that support other USB Type-C alternate modes (such as Thunderbolt 3), we strongly recommend they also support the following specifications in addition to ensure interoperability with all systems. These recommendations are in addition to the Windows Hardware Compatibility Specifications.

Expand table

Dock connection to the system	We recommend that the dock connect to the system by using USB Type-C. At a minimum, the dock's USB Type-C connector should support the following features: • USB Data (USB 3.1 Gen 2) to enable USB-connected peripheral devices • USB Power Delivery (PD) power provider for charging a system while it is docked, with a minimum power of 30W, following the PD power rules (see section 10 of the Power Delivery specification); for additional information, see the "Power budget policy" section

	• USB Type-C DisplayPort Alternate Mode, pin assignments C and D, with support for HBR and HBR2 signaling on those pin assignments (and pin assignments E and F, if supported), and the ability to sink DisplayPort on at least two DisplayPort lanes for all supported pin assignments If you plan on including an alternate mode other than DisplayPort over USB Type-C alternate mode, review the fallback requirements in the Overall recommendations section above.

Dock (downstream) I/O ports	The downstream device I/O ports between a dock and other peripheral devices should always interoperate with any system that meets the system docking requirements.
	• For peripheral device connectivity, a combination of at least three USB Type-A and/or USB Type-C 3.1 Gen 2 ports
	  ◦ The USB Type-C ports should provide a minimum of 15W to devices through USB Type-C current, and optionally USB PD; see the Power Budget Policy Section.
	  ◦ The USB Type-A ports should provide either 2.5, 4.5, or 7.5 W to devices according to the port type defined in the USB Battery Charging 1.2 standard; see the Power Budget Policy Section.
	• USB Type-C power input; see the "Power" recommendation
	• Ethernet connectivity (optional), to ensure that you choose a USB Ethernet adapter that works on mobile devices
	• Audio output port (optional)
	• An external monitor connection, if the dock does not include a built-in monitor (see the "Display output" recommendation)

Power	For the power input to the dock, we have the following recommendations:
	• The power input port should be a USB Type-C port.
	• If the power input port is a USB Type-C port and cannot be used to connect to the host system or to other devices, it should be clearly marked so that the user can easily determine that it is the power input port.
	• If the power adapter that ships with the dock is a USB Type-C power adapter, it must follow the PD Power Rules (see Section 10 of the Power Delivery specification). For example, a 36W supply must support the following contracts: 15V/2.4A, 9V/3A, and 5V/3A.
	• The power adapter that ships with the dock should budget enough power to charge the system connected to the dock along with other downstream devices connected to the dock, according to the following guidelines:
	  ◦ For the upstream USB-C host, budget a minimum of 30W (15V/2A).
	  ◦ For each downstream USB-C port, budget 15W (5V/3A).
	  ◦ For each downstream USB-A 2.0 port, budget 2.5W (5V/0.5A).
	  ◦ For each downstream USB-A 3.0 port, budget 4.5W (5V/0.9A).
	  ◦ For each downstream USB-A charging downstream port, budget 7.5W (5V/1.5A).
	
	For example, if a dock has a USB-C port for the host (30W), two USB-A 3.0 ports (2 x 4.5W), and a USB-A charging downstream port (7.5W), the

	power adapter should budget at least 46.5W. • The dock's upstream Type-C port that connects to the host should be capable of performing data and power role swaps, and it should request to be a PD power source when it is connected to the host. • The dock's upstream Type-C port that connects to the host should support fast role swap so that functionality provided by connected devices is not interrupted when external power is connected or disconnected.
Power budget policy	A dock's power budget is the total amount of power that the dock can provide to the connected host and connected devices. The dock can receive its power budget from a combination of internal sources and external sources (for example, a battery and a power adapter, respectively). A power budget policy is required to allocate power in a consistent manner if the total power requirements of the connected host and connected devices exceeds the dock's power budget, such as if a 30W charger is connected to the dock that is connected to a PC that is requesting 27W, while a device is requesting 15W. The following are recommendations for the power budget policy: • If a host is charging from the dock and a device is connected that would cause the dock's total power budget to be exceeded, the dock should determine whether the amount of power needed by the connected device can be satisfied by reducing the amount of power provided to the host. 1. If the answer to the question is yes, reduce the amount of power provided to the host by the amount needed to meet the power requirements of the connected device, and notify the host accordingly. ◦ The PD power rules may cause the voltage offered to the host to be reduced and/or cause the host to stop charging from the dock altogether. ◦ Later, if the dock's power budget is increased, or if another device is disconnected allowing the dock to reclaim some power, the dock should allocate that power back to the host and notify it accordingly. 2. If the answer to question is no, do not provide any power to the connected device. Later, if the dock's power budget is increased, or if another device is disconnected allowing the dock to reclaim some power, the dock should redetermine whether it can allocate sufficient power to the connected device that was not initially powered. • If a host is not already charging from the dock and charging the host would exceed the dock's total power budget, the dock should allow the host to enumerate the dock and the connected devices on the dock as normal, and the dock should keep providing power to the connected devices instead of charging the host. Later, if the dock's power budget is increased, or if another device is disconnected allowing the dock to reclaim some power, the dock should redetermine whether it can allocate sufficient power to charge the host.

	• In general, docks should avoid sinking power from hosts. However, if the dock is not connected to external power, it can attempt to sink power from the host so that its connected devices can function.
Display output	The connection from the dock to an external monitor should be provided by at least one digital connector that is at a minimum either HDMI 1.4 or DisplayPort 1.2 (supporting DP++). However, if the device is a docking display (that is, the dock and the monitor are in the same device), it is optional to include an additional display output to enable a multimonitor experience. The resolution supported should at a minimum be 1080p, regardless of whether the display is externally connected through an external video port or is internally connected to the display. If you plan on including an alternate mode other than DisplayPort, review the fallback requirements in the Overall recommendations section above.
Audio output	If the dock has one or more 3.5 mm audio output jacks, a minimum of stereo outputs need to be supported at 16 bits / 44.1 kHz format. Audio support in the dock must comply with the USB Audio Device Class 2 specification and implement jack insertion detection as outlined in the specification.
Audio/video latency	The audio and video latency of devices attached either through USB (or for audio through a 3.5 mm jack) should not exceed 40 milliseconds.
Touchpad (optional)	If the dock has an embedded touchpad, the touchpad needs to be a precision touchpad (PTP). We recommend that IHVs follow the PTP implementation guide to create PTPs that are on par with traditional laptop PTPs. The Device Integration, Experience Customization, and Module Design for HLK Requirements sections in the implementation guide describe the best practices for creating a touchpad that is superior aesthetically and experience-wise.

All these recommendations are in addition to the touchpad requirements detailed in the minimum hardware requirements.

Wireless docking

Like a wired dock, a wireless dock has ports for input and output peripherals. The connection to the device is wireless only. The dock does not provide power or a wired network connection to the device.

Wireless docking features to consider:

- A wireless connection from the device to the dock
- Support for phone, tablet, and laptops
- Wired or wireless mouse and keyboard input
- Monitor outputs using HDMI, DisplayPort, or both

Wireless dock discovery

Windows 10 will discover, pair, connect, and manage docks.

Recommendations

- Include Wifi-Display (Miracast) extensions in receiver firmware.
- Support WSB and MA-USB.
- Support WiGig for enterprise class docking equipment.

Related topics

Windows support for USB Type-C connectors

Continuum

Tablet mode

Article • 03/18/2021

Overview

Tablet mode is a new, adaptive user experience offered in Windows 10 that optimizes the look and behavior of apps and the Windows shell for the physical form factor and customer's usage preferences. This document describes how to implement tablet mode on 2-in-1 devices and tablets, specifically how to switch in and out of "tablet mode."

Tablet mode is a feature that switches your device experience from tablet mode to desktop mode and back. The primary way for a user to enter and exit "tablet mode" is manually through the Action Center, by clicking the icon in the lower right on the taskbar. In addition, OEMs can report hardware transitions (for example, transformation of 2-in-1 device from clamshell to tablet and vice versa), enabling automatic switching between the two modes. However, a key promise of tablet mode is that the user remains in control of their experience at all times, so these hardware transitions are surfaced through a toast prompt that must be confirmed by the user. Users also have the option to set the default response.

Target Devices

Tablets	**Detachables**	**Convertibles**
Pure tablets and devices that can dock to external monitor + keyboard + mouse.	Tablet-like devices with custom designed detachable keyboards.	Laptop-like devices with keyboards that fold or swivel away.

When the device switches to tablet mode, the following occur:

- Start resizes across the entire screen, providing an immersive experience.
- The title bars of Microsoft Store apps auto-hide to remove unnecessary chrome and let content shine through.

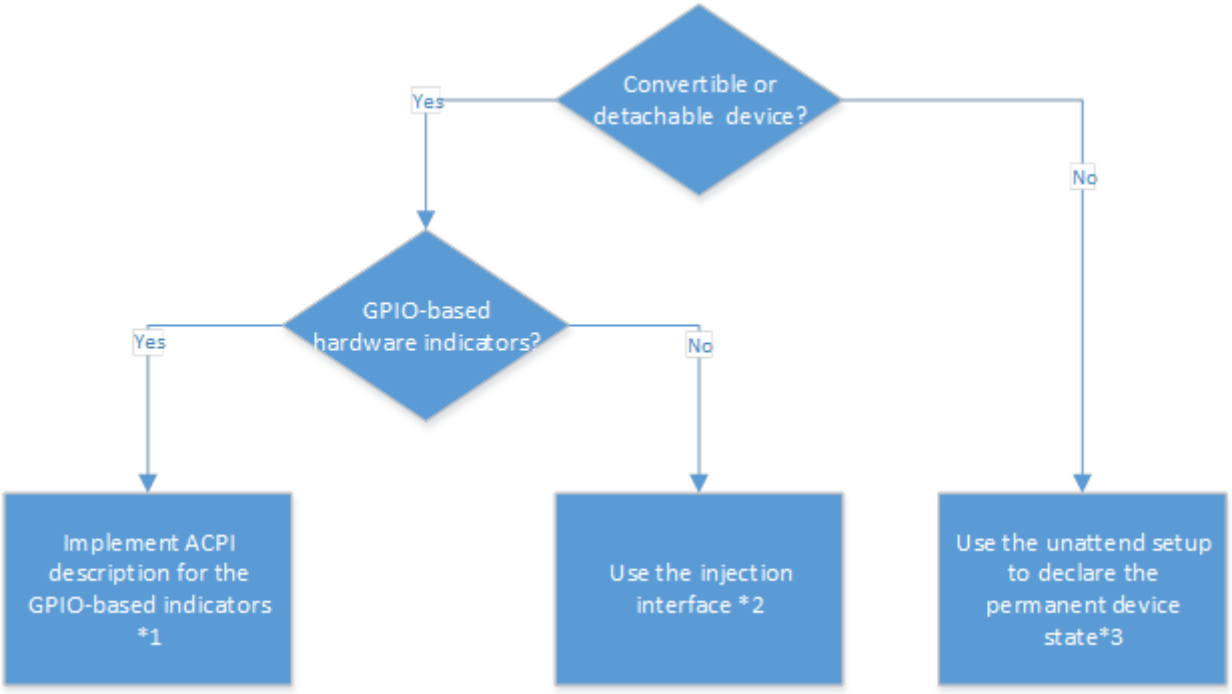
- Microsoft Store apps and Win32 apps can optimize their layout to be touch-first when in tablet mode.
- The user can close apps, even Win32 apps, by swiping down from the top edge.
- The user can snap up to two apps side-by-side, including Win32 apps, and easily resize them simultaneously with their finger.
- Apps can auto-launch side-by-side to create multitasking layouts without much user effort.
- The taskbar transforms into a navigation and status bar that's more appropriate for tablets.
- The touch keyboard can be auto-invoked.

Of course, even in "tablet mode", users can enjoy Windows 10 features such as Snap Assist, Task View and Action Center. On touch-enabled devices, customers have access to touch-friendly invocations for those features: they can swipe in from the left edge to bring up Task View, or swipe in from the right edge to bring up Action Center.

Tablet mode gives customers the flexibility to use their device in a way that is most comfortable for them. For example, a customer might want to use their 8" tablet in "tablet mode" exclusively until they dock it to an external monitor, mouse, and keyboard. At that point the customer will exit "tablet mode" and use all their apps as traditional windows on the desktop—the same way they have in previous versions of Windows. Similarly, a user of a convertible 2-in-1 device might want enter and exit "tablet mode" as they use their device throughout the day (for example, commuting on a bus, sitting at a desk in their office), using signals from the hardware to suggest appropriate transition moments.

Designing your device for tablet mode

If you are designing a 2-in-1 device that is a convertible or detachable, we recommend you go through the following flow chart when designing your product. This will ensure you have the right implementation in place to allow the tablet mode prompt to show when the device's state changes. Each of the boxes in the flow chart contain an implementation option.



Method 1 - Implement ACPI description for the GPIO-based indicators

If your system is a 2-in-1 device that uses physical GPIOs to indicate state transitions, you can describe the state transition indicator in ACPI and Windows will automatically forward the GPIO interrupts fired by that indicator up to the operating system.

For more information, review the following resources on MSDN.

- Indicator implementation (https://msdn.microsoft.com/library/windows/hardware/dn457884.aspx)
- ACPI descriptor samples (https://msdn.microsoft.com/library/windows/hardware/dn457868.aspx)

Method 2 - Use the injection interface

If your system is a 2-in-1 device that does not use physical GPIOs to indicate state transitions, then you must develop either a user or kernel mode driver that injects the state transition indication into Microsoft's "GPIO Laptop and Slate indicator driver".

The injection interface to the "GPIO Laptop and Slate indicator driver" is defined as the following:

```
/* 317fc439-3f77-41c8-b09e-08ad63272aa3 */
DEFINE_GUID(GUID_GPIOBUTTONS_LAPTOPSLATE_INTERFACE, 0x317fc439, 0x3f77, 0x41c8,
0xb0, 0x9e, 0x08, 0xad, 0x63, 0x27, 0x2a, 0xa3);
```

Calling WriteFile against the interface allows you to toggle the state of the indicator. A code sample for how a driver would use the injection interface can be found on the MSDN page here:
https://msdn.microsoft.com/library/windows/hardware/dn457889.aspx

If you are creating a driver that uses the injection interface, you will still need to declare the ACPI description for the laptop/slate mode indicator to ensure Microsoft's "GPIO Laptop and Slate indicator driver" loads. However, you will not need to add a GPIO resource to the ACPI declaration for the "laptop and slate indicator".

Note When the device boots, the "GPIO Laptop and Slate indicator driver" first initializes in "slate" mode. Thus, it's important for your injection driver to check the operating system state to ensure that it is in sync with the system's actual state immediately after the device boots.

Your SOC IHV partner may already offer a driver or other solution that injects the state transition into Microsoft's inbox driver. If that's the case, you can save time and effort by leveraging your IHV's solution according to their particular guidelines.

Method 3 - Use unattend setting to declare the permanent device state

If your device is not a 2-in-1 device, for example a regular laptop or desktop, you can disable prompting using the ConvertibleSlateModePromptPreference setting in your answer file. Set the value to 0 and the prompt will not show up and the UI will not switch. If no value is specified, the default value is 1, where the system will be prompted to switch modes.

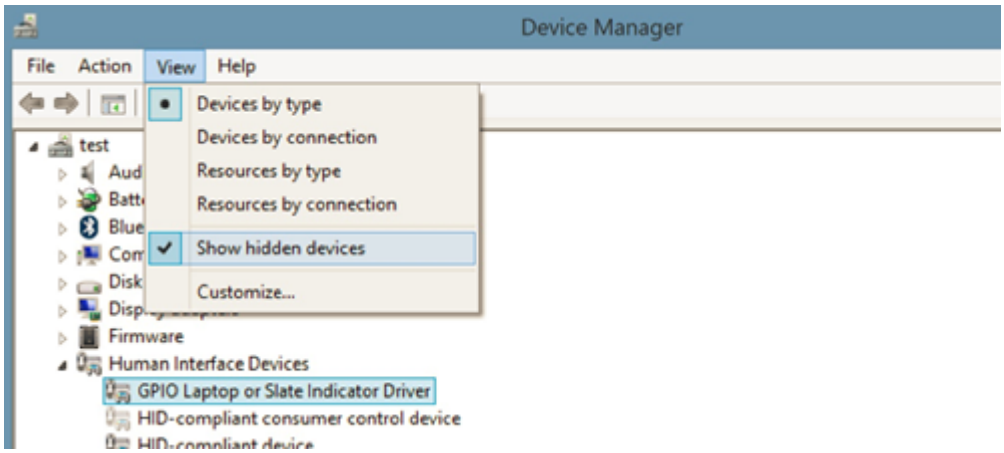
Note It is also recommended that you specify the type of form factor you are building using the DeviceForm setting in your answer file.

Test your device

Verify your device implements ConvertibleSlateMode

If you are unsure if your device correctly implements ConvertibleSlateMode, check if the "GPIO Laptop or Slate Indicator Driver" exists in the Device Manager.

Make sure to enable the “Show hidden devices” from the Device Manager to confirm the presence of this driver:



There are two main ways to enter/exit “tablet mode”:

- **Manually** - The user can enter/exit “tablet mode” manually by pressing the “tablet mode” button located in Action Center.
- **Hardware action triggered** - The OS can ask the user if they would like to enter/exit “tablet mode” if the hardware has indicated the form factor has changed through ConvertibleSlateMode.

Using the presence of a Bluetooth keyboard as a tablet mode trigger is not supported. The primary issue with using Bluetooth as a trigger is that the Bluetooth events correlate to device pairing and proximity and those events do not reliably signal whether the keyboard is in use (for example, it may not even be near the customer) and thus do not reliably signal customer intent to enter or exit “tablet mode”.

If you are interested in making sure that a device that ships with a Bluetooth-enabled keyboard provides hardware-based hinting for the customer to change modes, we recommend implementing a sensor that would understand the current form factor and then signal to the OS through ConvertibleSlateMode.

Because the Windows prompt to enter/exit “tablet mode” is dependent on the implementation of ConvertibleSlateMode, it is important to ensure the firmware that

toggles ConvertibleSlateMode is tested thoroughly. ConvertibleSlateMode implementations of lower quality can lead to a poor end user experience, such as asking the user to "enter 'tablet mode'" multiple times.

In addition, when a 2-in-1 device changes from clamshell to tablet, the keyboard and touchpad can receive accidental presses from the user holding the tablet. Windows does not support ignoring input from the built-in keyboard and touchpad when a convertible changes into a tablet. We expect that device manufacturers do work in their hardware or software to avoid unintentional input during the form factor change, and test it thoroughly. The following table shows a few of the common errors that can occur, along with use cases that could cause them. These use cases should be tested to prevent the user from having a poor switching experience:

Error	Use cases that can cause the error
ConvertibleSlateMode switches even though there is no customer-observable reason (for example, a false firing).	• Mechanical/electrical issues • Technical reasons causing the switch to occur (for example, having to power down a component early as the system goes to sleep, causing a spurious transition)
ConvertibleSlateMode thinks that the system is currently a slate form factor, when it's actually a laptop form factor (or vice versa).	• Changing the device's form factor while the device is in a power state transition (for example, powering up/down, going to sleep, etc). • Changing the system's form factor very slowly • Changing the system's form factor very quickly
ConvertibleSlateMode switches multiple times in rapid succession, causing the OS to repeatedly prompt the user to enter/exit "tablet mode".	• Changing the system's form factor very slowly • Changing the system's form factor very quickly

Unattend Settings

The following unattend settings are useful if you are designing for a tablet mode-capable device:

Unattend Setting	Description	Value

Unattend Setting	Description	Value
ConvertibleSlateMode	With the ConvertibleSlateMode setting, you specify the physical ability of the device, which impacts physical behaviors of the device (such as rotation and onscreen keyboard), and you help define the appropriate first use experience that will be presented to an end user. If you specify laptop mode, an end user can type on a physical keyboard. If you specify tablet mode, an end user cannot type on a physical keyboard, because the keyboard is removed or the keyboard is flipped back so that it is inaccessible.	• 0 – Tablet, convertible, or detachable (default) • 1 – Laptop
ConvertibleSlateModePromptPreference	With the ConvertibleSlateModePromptPreference setting, you can specify whether to prompt the user to enter or exit tablet mode. The device form factor or product design will influence whether you choose to show the prompt. If you choose to hide the prompt, the mode is not switched. (That is, if tablet mode is on, it stays on, and if tablet mode is off, it stays off.) If you want the prompt to appear when the hardware event occurs, you need to implement a hardware or software solution to toggle this setting.	• 0 – Disabled; the prompt does not appear, and the UI does not switch modes • 1 – Enabled (default); the prompt appears, and the UI switches modes if the user consents

Unattend Setting	Description	Value
SignInMode	You use the SignInMode setting to specify whether to set the display to tablet mode or to desktop mode after a user logs on. If you do not explicitly set this value, the default is tablet if the screen size is less than 10 inches and your preferred power management profile (power platform role) is slate (that is, a tablet form factor). Otherwise, the default is the last setting used. If there is no previous setting, the last setting used is considered to be desktop.	• 0 – Tablet (default if the screen size is less than 10 inches and the power platform role is slate) • 1 – Desktop • 2 – Last setting used (default if the tablet default requirements are not met); if there is no actual last setting, desktop is considered the last setting
DeviceForm	You specify the DeviceForm setting so that a variety of apps (for example, Bing and Cortana) can determine the correct content for the specified device form factor. You also use this setting for features such as retail demo mode to specify the most suitable demo content for the device type.	There are 30 possible values. The primary values are listed below: • 0 – Unknown (default) • 2 – Tablet • 4 – Laptop • 5 – Convertible • 6 – Detachable

Behavior of "tablet mode"

To achieve a great tablet experience in Windows, the following desktop aspects of the taskbar are optimized for tablet mode:

- **Showing running apps** - this suggests to people that they need to manage their running apps' life cycle.
- **Several entry points to the Desktop** - a tablet user does not need easy access to the desktop in "tablet mode."
- **A dense layout in the taskbar** - this can fit more items but in "tablet mode," touch is likely the primary input, and requires a less dense layout.
- **Showing Deskbands and Win32 notification icons at the top level and the overflow level** - the majority of these are not intentionally placed there by the user and are not commonly used.

Rotation lock

Rotation lock is set to enabled (or locked) by default. Unless OEMs set the rotation lock to unlocked, customers unboxing new tablets might be frustrated by OOBE not rotating to landscape or portrait according to how they hold their new devices. By disabling rotation lock, OEMs can make sure customers are delighted with the responsiveness of their new devices, even before they explore the ease of managing the rotation lock setting in the Quick Actions area of the Action Center.

To set rotation lock to disabled and enable rotation by default:

1. In Audit mode, set **HKLM\Software\Microsoft\Windows\CurrentVersion\AutoRotation\Enable** to 1, where 1 means rotation is enabled.
2. Use Sysprep to prepare the machine for the customer as you would normally.

Notification area icons

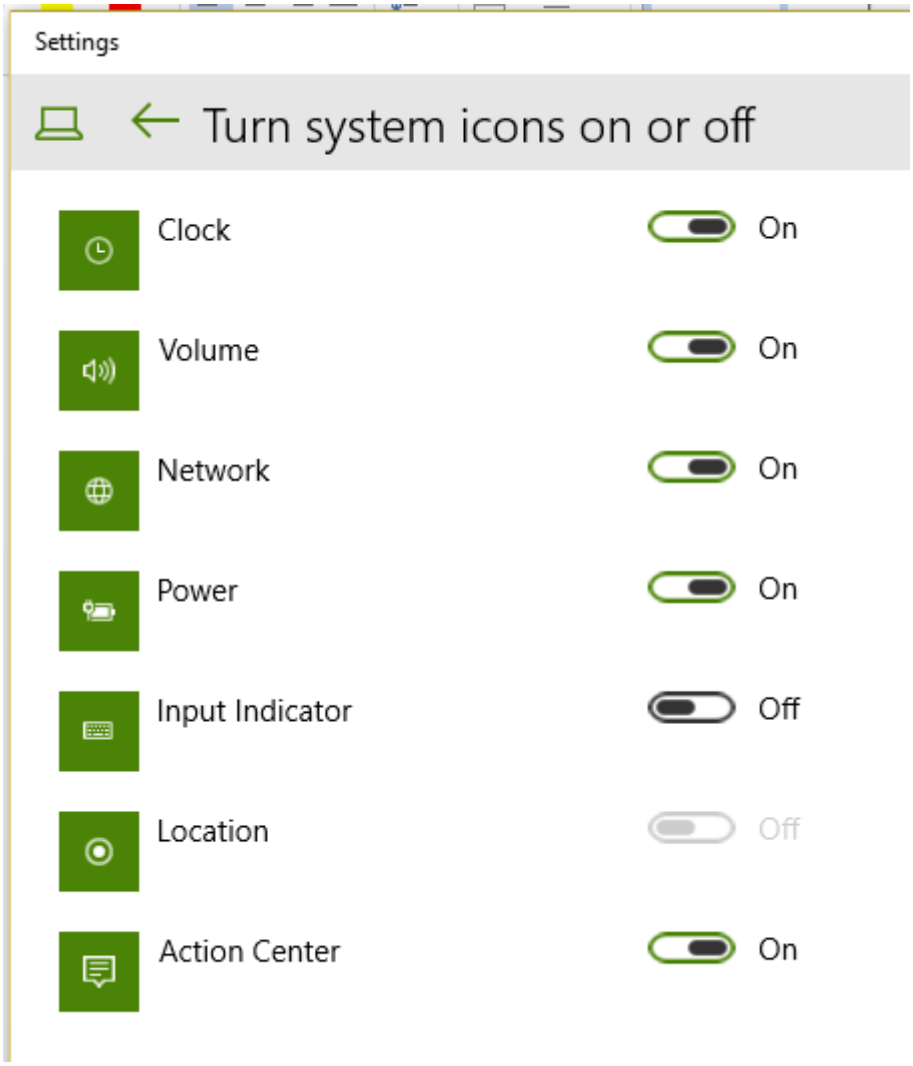
When entering tablet mode, the notification area will go through several changes. The following icons will be shown:

- Battery
- Network
- Volume
- Action Center

More specifically, if the user has demoted any of these four icons into the overflow, it will be brought back to the top level, or if the OEM has demoted the battery icon into the overflow, it will be brought back to the top level.

However, if the customer has disabled one of these in Settings under **Turn system icons on or off** (shown below), Windows will follow the customer's manual setting and not

enable the icon.



OEM-pinned notification icons will be hidden, user-pinned notification icons will be hidden, and the overflow chevron will be hidden, effectively hiding the overflow window.

Auto-promotion of notification icons to the top level will be blocked, but apps should continue to be able to fire a toast notification, even if the icon cannot come to the top level. The one exception to this is the location services icon. When it is enabled under **Turn system icons on or off**, it will be able to promote itself to the top level, to the left of the battery icon. However, if the user has manually chosen to disable the location services icon using the **Turn system icons on or off** setting, Windows will not force the icon to promote.

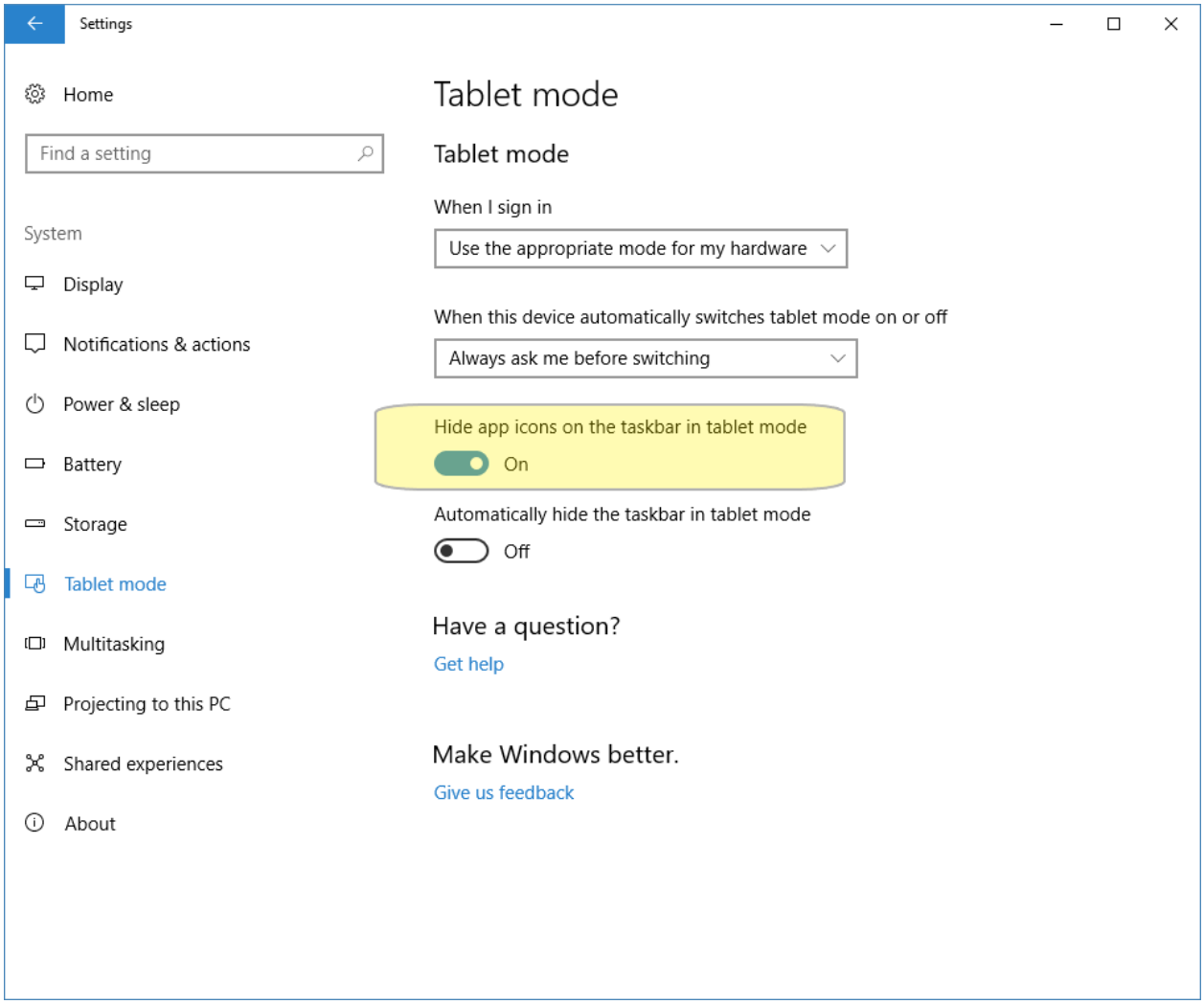
If customers select the **Always show all icons in the notification area** setting, the above behavior will turn off, and the notification area will revert to the same contents and behavior as if tablet mode were not being used.

Touch keyboard button

When entering tablet mode, the Touch Keyboard button will be hidden. Its visibility can be controlled via the **Show touch keyboard button** setting in the tablet mode taskbar context menu.

App icons

By default, app icons will be hidden on the taskbar in Tablet Mode. This is controlled by the **Hide app icons on the taskbar in tablet mode** setting:



Creating an app that interacts with tablet mode APIs

You can take advantage of tablet mode by using Universal Windows Platform (UWP) and Classic Windows APIs in your apps. Using these APIs, you can tailor the user experience of your app to be touch-first for tablet mode or mouse-first for desktop usage. This section describes how to use both the UWP and Classic Windows APIs in your applications to detect and react to tablet mode.

The APIs allows both Windows desktop applications and Universal Windows apps to query for the current interaction mode and respond to changes to the mode, as described here.

Classic Windows apps APIs

UIViewSettings	Allows apps to query the current pointer device preference state of the system for a particular application. It is a non-activatable class that you get from GetForCurrentView.
UserInteractionMode	Property of GetForWindow that takes a value in the Windows.UI.ViewManagement.UserInteractionMode enum, the allowed values of which are Touch and Mouse.
WM_SETTINGSCHANGE	Apps are notified when the current input state changes by broadcasting WM_SETTINGSCHANGE with “UserInteractionMode”.

Universal Windows apps APIs

UIViewSettings	Allows apps to query the current pointer device preference state of the system for a particular application. It is a non-activatable class that you get from GetForCurrentView.
UserInteractionMode	Property of UIViewSettings that takes a value in the Windows.UI.ViewManagement.UserInteractionMode enum, the allowed values of which are Touch and Mouse.
SizeChanged	Event is triggered when the current input state changes. Apps should query for this event.

Example 1: Optimize layout on app launch

When an app launches it can query for the current state of the system and optimize its layout for the current interaction mode, as in the following example.

Universal Windows App (C#) Sample

```
using Windows.UI.Xaml;
using Windows.UI.ViewManagement;

    public sealed partial class MainPage : Page
    {
        public MainPage()
        {
            InitializeComponent();
            // Every view gets an initial SizeChanged, so we will do all our
            // work there. This means that our view also responds to dynamic
            // changes in user interaction mode.
```

```
            Window.Current.SizeChanged += SizeChanged;
        }

        private void SizeChanged(object sender, RoutedEventArgs e)
        {
            switch(UIViewSettings.GetForCurrentView().UserInteractionMode)
            {
                case UserInteractionMode.Mouse:
                    VisualStateManager.GoToState(this, "MouseLayout", true);
                    break;

                case UserInteractionMode.Touch:
                default:
                    VisualStateManager.GoToState(this, "TouchLayout", true);
                    break;
            }
        }
     }
```

Classic Windows App (C++) Sample

```
using namespace Microsoft::WRL::Wrappers;
using namespace Microsoft::WRL;
HRESULT OnLayout()
{
    ComPtr<IUIViewSettingsInterop> uiViewSettingsInterop;
    HRESULT hr = GetActivationFactory(

HStringReference(RuntimeClass_Windows_UI_ViewManagement_UIViewSettings).Get(
),
        &uiViewSettingsInterop);
    if (SUCCEEDED(hr))
    {
        ComPtr<IUIViewSettings> uiViewSettings;
        hr = uiViewSettingsInterop->GetForWindow(hwndApp,
IID_PPV_ARGS(&uiViewSettings));
        if (SUCCEEDED(hr))
        {
            Windows::UI::ViewManagement::UserInteractionMode mode;
            hr = uiViewSettings->get_UserInteractionMode(&mode);
            if (SUCCEEDED(hr))
            {
                switch (mode)
                {
                case UserInteractionMode_Mouse:
                    hr = LayoutForMouse();
                    break;

                case UserInteractionMode_Touch:
                default:
                    hr = LayoutForTouch();
```

```
                    break;
                }
            }
        }
    }
    return hr;
}
```

Example 2: React to tablet mode entry/exit

When your device enters or exits tablet mode, the shell prompts the user to switch modes. You can use this as a trigger to optimize your app for the current interaction mode. You must query for the current state of the system, as in the following examples.

Universal Windows App (C#) Sample

Same as Example 1.

Classic Windows App (C++) Sample

```
// WM_SETTINGCHANGE handler
void OnSettingsChange(LPARAM lParam)
{
    if (lParam != 0)
    {
        PCWSTR message = reinterpret_cast<PCWSTR>(lParam);
        if (CSTR_EQUAL == CompareStringOrdinal(message, -1,
L“UserInteractionMode”, -1, TRUE))
        {
             OnLayout();
        }
    }
}
```

Frequently Asked Questions (FAQ)

#	Question	Answer
1	Will any of the file association change when the device is in “tablet mode”?	No, Windows will not change any file associations as all apps are supported in “tablet mode”.
2	Will the auto prompt to enter/exit “tablet mode” take place upon the insertion of an external USB keyboard?	No, the auto prompt will be triggered through the toggling of the ConvertibleSlateMode value, not the presence of any kind of keyboard.

#	Question	Answer
3	What triggers the "GPIO Laptop or Slate Indicator driver" to show up in Device Manager?	Defining PNP0C60 (laptop/slate mode state indicator) will trigger the driver to show in Device Manager. You must choose “Show the hidden device” option from Device Manager to show this driver.
4	Can we use pure All-in-ones (AiO) with external keyboard and mouse to enable the same scenario?	Yes, AiOs can take advantage of “tablet mode” by implementing ConvertibleSlateMode toggling at the appropriate transition point for the hardware.
5	Is touch keyboard auto disabled while “tablet mode” is disabled?	The touch keyboard only auto-invokes while in “tablet mode”. During Desktop usage, the touch keyboard must be invoked manually. If a user is not in "tablet mode", but they remove their keyboard from the device, the touch keyboard will appear.
6	Will a Bluetooth keyboard be considered as a “dock”?	No, a Bluetooth keyboard will not be considered a “dock” as that scenario may not always be true. Tablet mode never uses the presence of a keyboard (USB, Bluetooth, etc.) to trigger a prompt. You can implement a Bluetooth detachable design by toggling ConvertibleSlateMode at the appropriate time.
7	Win32 apps: For a Win32 app to take advantage of tablet mode, does the app need to be compiled with Universal Windows Platform (UWP) APIs?	Win32 apps will have a Win32-specific API that they can consume to adapt between touch-first and mouse-first usage.
8	For an application developer, do we need to call any API to handle tablet and desktop mode switching? Or will Windows handle it automatically?	If you want to change the layout inside your application, you will need to use either a UWP API or a Win32 API to indicate “tablet mode”. The actual resizing of the application when entering/exiting “tablet mode” is automatically handled.
9	My system implements ConvertibleSlateMode but state transitions don’t result in a prompt asking me to enter/exit “tablet mode”.	If your 2-in-1 device toggles ConvertibleSlateMode correctly, but the system does not prompt you to enter/exit “tablet mode” the feature may be turned off. To turn tablet mode on, go to **Settings\System\Tablet Mode**. From the dropdown list, select “When my device wants to switch modes” and select “Always prompt me to confirm.”

#	Question	Answer
10	For tablets, do we need a Virtual Driver if GPIO is supported? If GPIO is available do we need any BIOS changes?	You have a choice of using the physical GPIO or the injection driver approach. As long as you are properly toggling ConvertibleSlateMode, no changes are necessary. Important: Windows 10 implementation of ConvertibleSlateMode is different from Windows 8.1. You must thoroughly test your device to ensure correct behavior.
11	Is there requirements or changes to UEFI BIOS to support tablet mode?	There are no inherent changes to the UEFI BIOS required to support tablet mode. You are free to toggle ConvertibleSlateMode by whatever means work best for you.
12	Is screen rotation tied to tablet mode?	No, screen rotation is not necessarily related to tablet mode. Screen rotation is tied to your current device configuration which may or may not map to whether the device is in “tablet mode”.
13	What happens when the taskbar is located on the left side?	The taskbar stays where it was positioned.
14	Are there plans for enabling a scenario where the system would have tablet mode enabled or disabled automatically based on the laptop or slate mode reported by the system, without showing any confirmation to the end customer?	Not supported. We always want users to be in control and choose whether they want to switch the mode. Users can override this if they wish.
15	Will the UI that asks to "Enter tablet mode" be changed? Do you plan on adding a checkbox to let the user decide to "perform this action every time"?	Yes, it will change from the current design. It should include a “Yes/No” button and a dropdown list to let a user choose what to do next time (such as “Always prompt”, and more).
16	Can you show where the Global Back Button is in tablet mode?	It is next to the Start button.

Windows Hello

Article • 11/05/2020

Microsoft Windows Hello, part of Windows 10, gives users a personal, secured experience where the device is authenticated based on their presence. Users can log in with a look or a touch, with no need for a password. In conjunction with Microsoft Passport, biometric authentication uses fingerprints or facial recognition and is more secure, more personal, and more convenient.

In this section

Topic	Description
Windows Hello and the Companion Device Framework	Windows Hello works with the Companion Device Framework to enhance the user authentication experience. Using the Windows Hello companion device framework, a companion device can provide a rich experience for Windows Hello even when biometrics are not available (e.g., if the Windows 10 desktop lacks a camera for face authentication or fingerprint reader device, for example).
Windows Hello biometric requirements	Learn about the hardware requirements for biometric equipment, such as IR camera and fingerprint readers in order to support Windows Hello.
Windows Hello face authentication	Microsoft face authentication in Windows 10 is an enterprise-grade identity verification mechanism that's integrated into the Windows Biometric Framework (WBF) as a core Microsoft Windows component called Windows Hello. Windows Hello face authentication utilizes a camera specially configured for near infrared (IR) imaging to authenticate and unlock Windows devices as well as unlock your Microsoft Passport.

Related topics

Windows Biometric Framework API

Windows Hello Enhanced Sign-in Security

Article • 06/19/2024 • Applies to: ✅ Windows 11, ✅ Windows 10

Windows Hello enables biometrics or PIN authentication, eliminating the need for a password. Biometric authentication uses facial recognition or fingerprint to prove a user's identity in a way that's secure, personal, and convenient.

Enhanced Sign-in Security (ESS) provides an additional level of security to biometric data with the use of specialized hardware and software components. Virtualization Based Security (VBS) and Trusted Platform Module 2.0 are used to isolate and protect user's authentication data, and to secure the data communication channel.

How does Enhanced Sign-in Security protect biometric data

ESS and facial recognition

When ESS is enabled, the face algorithm is protected using VBS to isolate it from the rest of Windows. The hypervisor is used to specify and protect memory regions, so that they can only be accessed by processes running in VBS. The hypervisor allows the face camera to write to these memory regions providing an isolated pathway to deliver face data from the camera to the face matching algorithm.

Face templates are generated in VBS by the protected face algorithm. When not in use, face template data is encrypted using keys generated and only accessible to VBS, and then stored on disk.

ESS and fingerprint recognition

ESS is only supported on fingerprint sensors with match on sensor capabilities. This type of sensor includes a microprocessor and memory which can be used to isolate fingerprint matching and template storage using hardware.

Sensors that support ESS have a certificate embedded during manufacturing. The certificate can be validated by the Windows biometric components running in VBS, and is used to establish a secure session with the sensor. The sensor and Windows biometric components use the session to communicate enrollment operations and match results securely.

Credential operations

The Windows biometric components running in VBS establish a secure channel to the TPM using information shared with VBS by the TPM during boot. When a matching operation is a success, the biometric components in VBS use the secure channel to authorize the usage of Windows Hello keys for authenticating the user with their identity provider, applications, and services.

Enable Enhanced Sign-in Security

The enablement of ESS is dependent on specialized hardware, drivers, and firmware pre-installed on the system. Device manufacturers can choose to enable Enhanced Sign-in Security during device configuration in the factory.

> **Note**
>
> All Copilot+ PCs have ESS enabled by default. For more information, see Copilot+ PC hardware requirements.

System requirements

Compatible hardware and software components are required to enable Enhanced Sign-in Security:

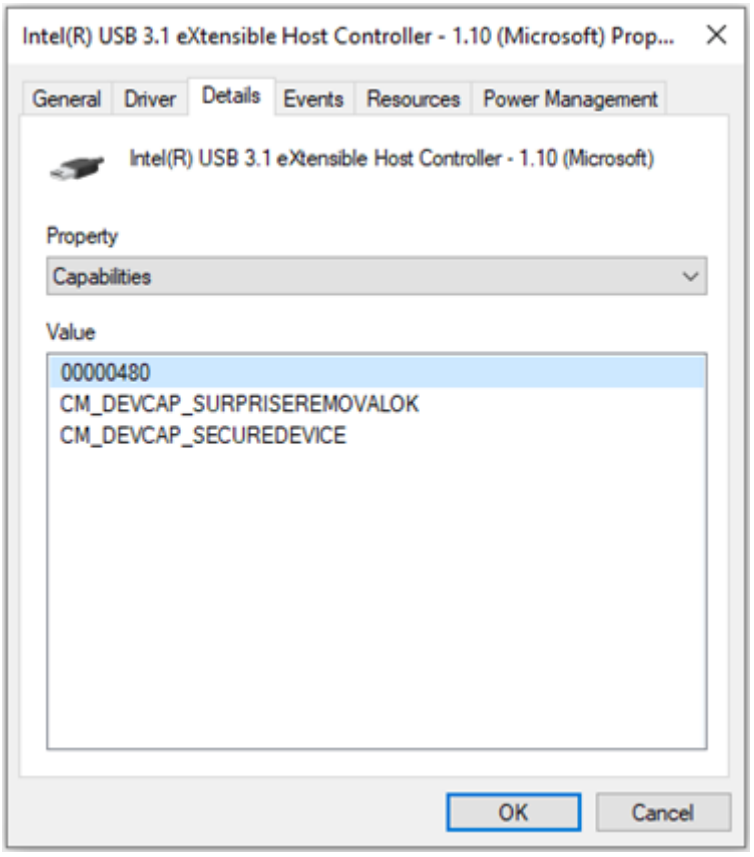
- Meet the requirements for Virtualization-Based Security (VBS), including Device Guard Enablement and Trusted Platform Module 2.0
- Biometric sensor hardware that supports ESS
- Biometric sensor drivers compatible with ESS
- Device firmware with a Secure Devices (SDEV) ACPI table configured by the device manufacturer for the included biometric hardware

Biometric sensor compatibility

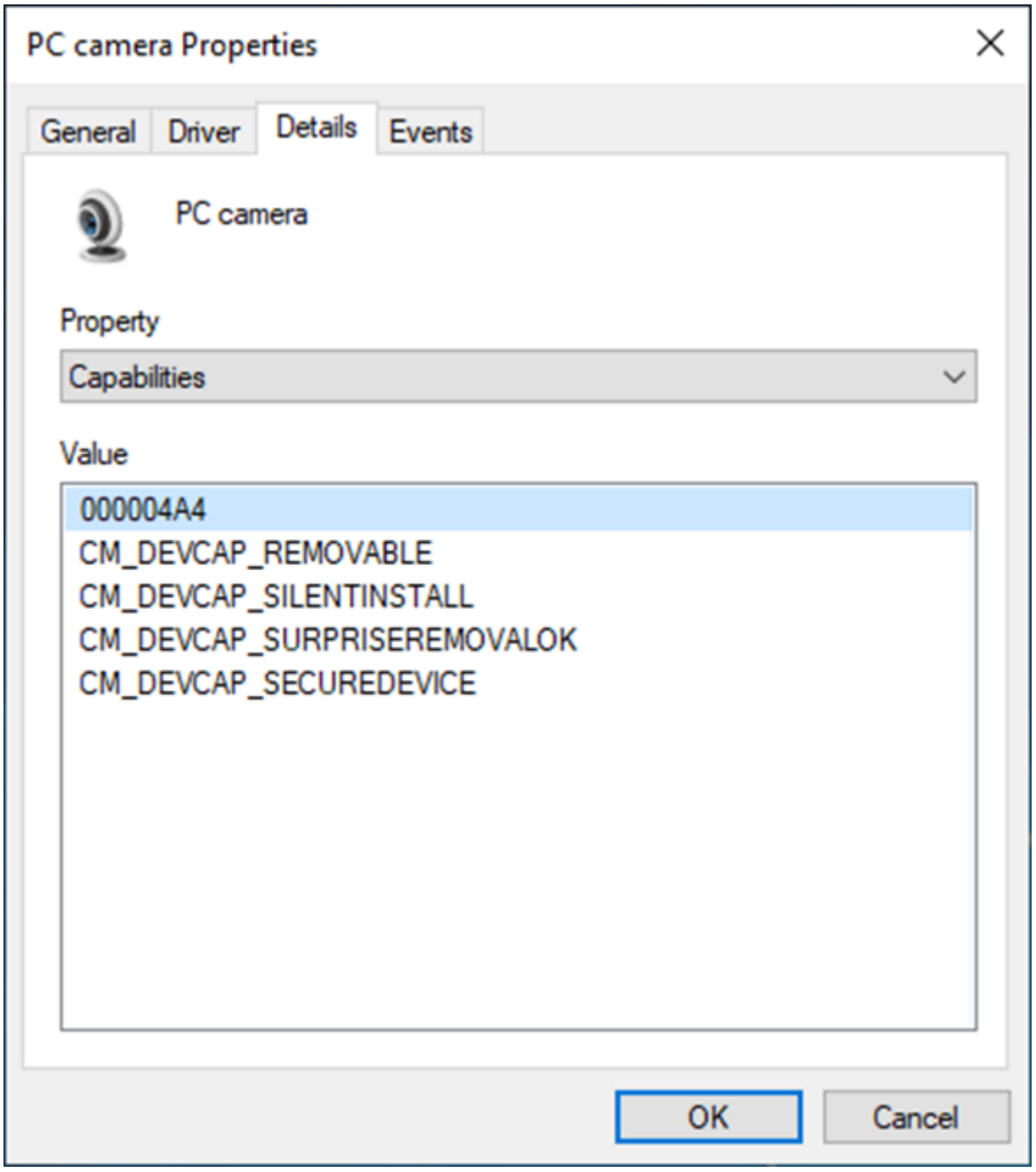
Face biometric sensor

ESS is designed to work with a select range of IR cameras, and it requires specific chipsets. Cameras that support ESS must have this feature built into their firmware, and using the standard Windows UVC camera driver that comes with the operating system is necessary.

To check if the camera module is ESS-capable, first go to Device Manager and expand the **Universal Serial Bus controllers** section. Right-click on the device that has **eXtensible Host Controller** in the name, and select the **Properties** option to view the device properties. If there are multiple entries for a host controller, check the properties section for all. Navigate to the **Details** tab of the driver and select **Capabilities** from the **Property** drop down menu. One of the devices should show the `CM_DEVCAP_SECUREDEVICE` capability.



Next, check the properties sections of the PC Cameras by going to the **Cameras** section in Device Manager. If there are multiple entries for PC Cameras, check the properties section for all. Navigate to the **Details** tab of the drivers and select **Capabilities** from the **Property** drop down menu. One of the PC camera devices should have the `CM_DEVCAP_SECUREDEVICE` capability.

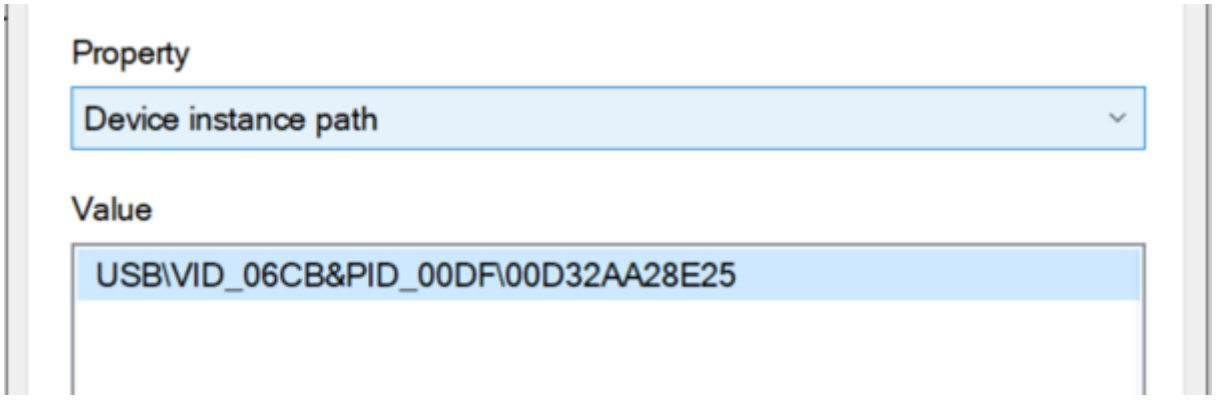


Fingerprint biometric sensor

ESS-capable fingerprint sensors must be match on chip:

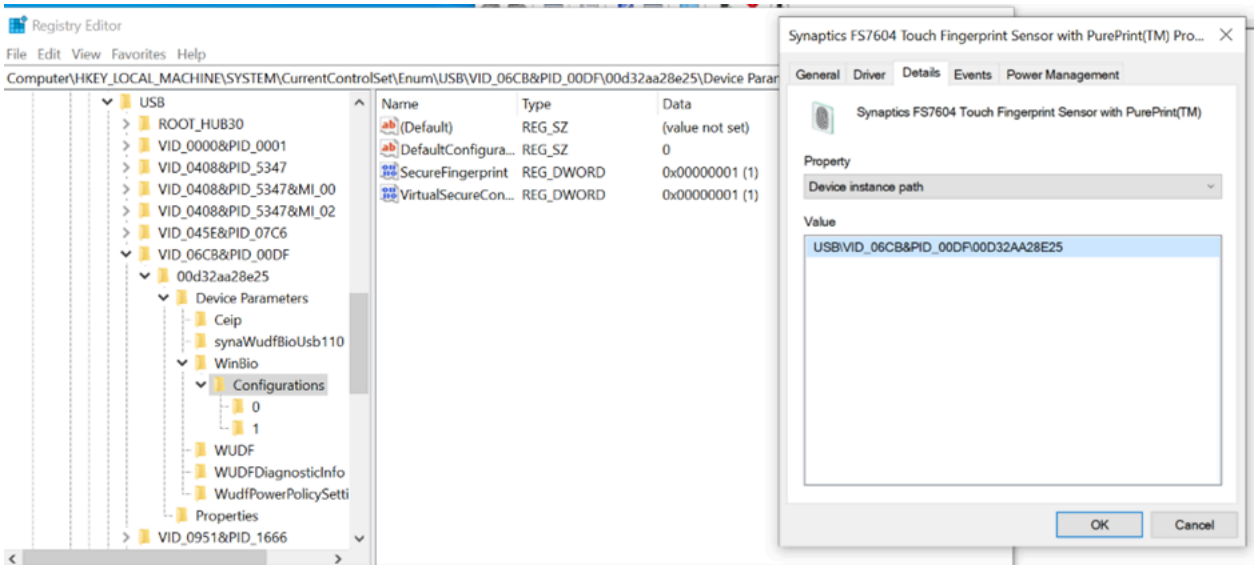
- The sensor must have a Microsoft-issued certificate burned into the device during manufacturing
- The device driver and firmware must support Enhanced Sign-in Security functionality

To check if a fingerprint module is ESS-capable, first go to Device Manager and expand the **Biometric devices** section. There should be an entry for a fingerprint sensor. Right-click the fingerprint reader entry and go to **Properties** > **Details**. Under the **Property** option, select **Device instance path**.



Open regedit.exe and navigate to `HKLM\SYSTEM\CurrentControlSet\Enum\[DeviceInstancePath]\Device Parameters\WinBio\Configurations` where `DeviceInstancePath` is the path listed in Device Manager. Select **Configurations**. There should be a registry key listed named `SecureFingerprint` with a data value of `1`. If it doesn't exist, the device isn't secure-capable.

Configurations should also have two folders beneath it: one labeled `0` and one labeled `1`. If there's only one folder and not two, the device isn't secure-capable.



Verify if ESS is enabled

Security Center

If ESS is enabled, the **Device Security** section of the Windows Security application has an entry for Enhanced Sign-in Security. The entry describes the hardware capability of the system. If the Enhanced Sign-in Security section isn't present, the feature isn't enabled on the system.

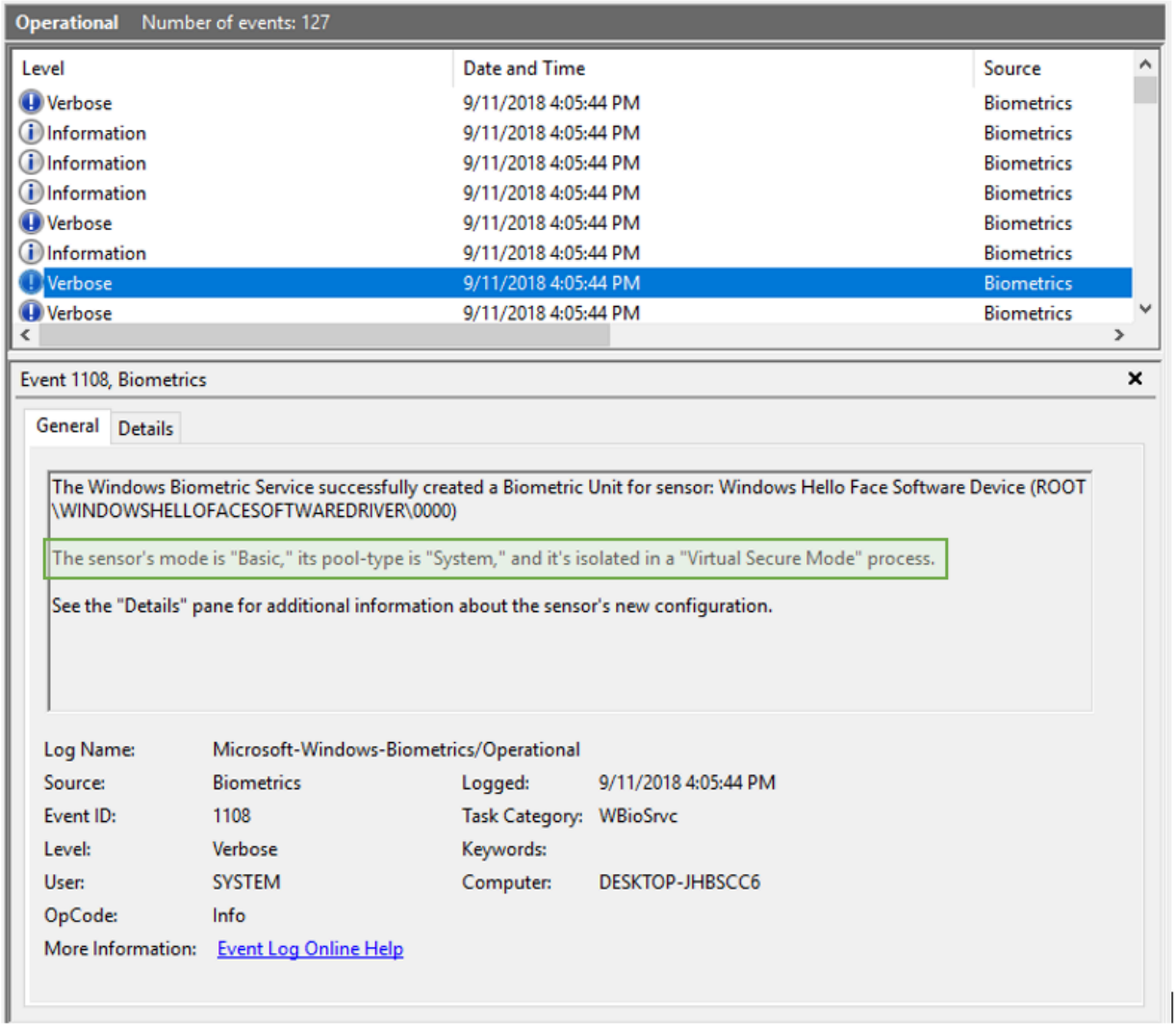
If there's a biometric sensor embedded in the device that doesn't support ESS, or that type of biometric hardware is absent from the system, it's indicated by the **Unavailable**

due to incompatible hardware description next to the corresponding sensor. This message indicates that the hardware doesn't follow the sensor requirements needed to support ESS.

Event Viewer

The Windows biometric framework generates logs events when each sensor on a system is enumerated. These logs include information indicating whether a sensor is operating with Enhanced Sign-in Security enabled. Biometric event logs are found in Event Viewer under **Event Viewer** > **Applications and Services Logs** > **Microsoft** > **Windows** > **Biometrics** > **Operational**.

If the biometric device is loaded properly by the Windows biometric framework, there's a log event ID `1108` for the corresponding sensor. If the device is operating with ESS enabled, the sensor is specified as *isolated* in a **Virtual Secure Mode** process. If the device isn't using ESS, it's specified as *isolated* in a **System** process.


Operational Number of events: 127
Level
Date and Time
Source
Verbose 9/11/2018 4:05:44 PM Biometrics
Information 9/11/2018 4:05:44 PM Biometrics
Information 9/11/2018 4:05:44 PM Biometrics
Information 9/11/2018 4:05:44 PM Biometrics
Verbose 9/11/2018 4:05:44 PM Biometrics
Information 9/11/2018 4:05:44 PM Biometrics
Verbose 9/11/2018 4:05:44 PM Biometrics
Verbose 9/11/2018 4:05:44 PM Biometrics
Event 1108, Biometrics
General
Details
The Windows Biometric Service successfully created a Biometric Unit for sensor: Windows Hello Face Software Device (ROOT\WINDOWSHELLOFACESOFTWAREDRIVER\0000)
The sensor's mode is "Basic," its pool-type is "System," and it's isolated in a "Virtual Secure Mode" process.
See the "Details" pane for additional information about the sensor's new configuration.
Log Name: Microsoft-Windows-Biometrics/Operational
Source: Biometrics
Logged: 9/11/2018 4:05:44 PM
Event ID: 1108
Task Category: WBioSrvc
Level: Verbose
Keywords:
User: SYSTEM
Computer: DESKTOP-JHBSCC6
OpCode: Info
More Information: Event Log Online Help

In the event `1108`, cameras are described using *Windows Hello Face Software Device (ROOT\WINDOWSHELLOFACESOFTWAREDRIVER\0000)* and fingerprint devices are

described using the device's specific module and device ID. For fingerprint devices, the device ID is listed in device Manager under **Biometric Devices** > **[Fingerprint Module]** > **Properties** > **Details** > **Device Instance Path**.

Application Compatibility

For devices with ESS-capable cameras, a Secure Devices (SDEV) table is required. When an SDEV table is implemented and VBS is turned on, the SDEV table is parsed by the Secure Kernel and restrictions are enforced on accessing Peripheral Component Interconnect (PCI) device configuration space. These restrictions are enacted to prevent malicious processes from manipulating the configuration space of secured devices specified in the SDEV table.

Applications that attempt to read/write the PCI configuration space, except by means explicitly supported by Windows, result in bug checks when the SDEV table is parsed and enforced.

All drivers and software included in the device image must be tested for compatibility, given these software restrictions. Software or drivers that are distributed to the system via Windows Update, the Microsoft Store, or other acceptable channels by the device manufacturer should also be checked for compatibility. Without this verification, there might be unexpected behavior on the system.

Unsupported scenarios

Non-ESS supported sensors

When ESS is enabled, only biometric sensors that support ESS work on the system. All noncapable sensors won't be enumerated by the Windows Biometric framework.

It's the manufacturer's decision on what hardware they include in the system and whether Enhanced Sign-in Security is enabled by default. If there are any concerns around biometric modalities being blocked, contact the device manufacturer for support.

Pluggable/peripheral biometric sensors

ESS isn't supported for external fingerprint sensors or camera modules. With ESS enabled, external, or peripheral biometric sensor operations are blocked, regardless of whether they're secure-capable or not. If you would like to use a peripheral with ESS to sign in with Windows Hello, see Disable/Enable ESS

Wake on touch for fingerprint sensors

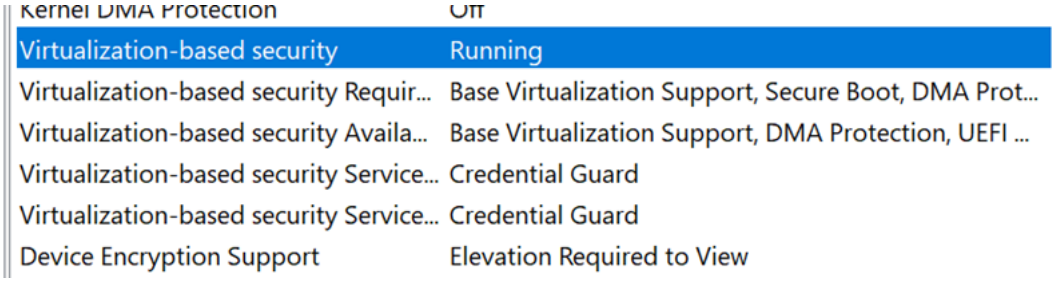
Wake on Touch (WoT) is the ability for the fingerprint sensor to wake the system and sign the user in without requiring the user to touch the sensor twice. Devices that support Modern Standby enable Wake on Touch sensor behavior.

Starting in Windows 11, version 22H2 with KB5027303, WoT is available for ESS devices.

Troubleshooting

Face/Fingerprint authentication isn't working

If biometric authentication isn't working, first check that VBS is running and that the secure component started. To check if VBS is running, open **System Information** > **System Summary**. There should be an entry for **Virtualization Based Security** listed as *Running*.



Also check that the biometric isolation trust-lets are running. These should be listed under **System Information** > **Software Environment** > **Running Tasks** as `bioiso.exe` and `ngciso.exe`. If either of these checks fail, the system might not meet the requirements for Enhanced Sign-in Security. Attempt to restart the biometric service using step #3.

1. In **Settings** > **Sign-in Options**, remove the nonfunctioning enrollment and re-enroll
   a. if the entry for Windows Hello Face/Fingerprint is unavailable with the condition *We couldn't find a fingerprint scanner compatible with Windows Hello Face*, or something similar, proceed to the next step
2. In device manager, the sensor should be listed under **Biometrics devices**. Reinstall the driver by right clicking the name of the device and select **Uninstall Device**. Restart the device, at which point Windows attempts to reinstall the driver. Check if authentication is working
3. To restart the biometric service, first remove PIN from the system by going to **Sign-in Options** and removing PIN. Open a command prompt as administrator and

enter `net stop wbiosrvc && net start wbiosrvc`. Check if fingerprint authentication is working

4. If biometrics is still not working on the device, file a feedback item using Feedback Hub

To check that the secure connection succeeded, refer to the Verify if ESS is enabled section.

PIN isn't working

PIN can be reset in the lock screen under **Sign-in options**. To do so, remove PIN and add it again. This will prompt PIN reset, which should restore PIN functionality.

Disable/Enable ESS

Starting in Windows 11, version 22H2 with KB5031455, users can temporarily turn off ESS if they would like to use an external peripheral to authenticate with Windows Hello on their device.

You can use the Settings app to disable ESS. Select **Start** > **Settings** > **Accounts** > **Sign-in options** or use the following shortcut:

Sign-in options

Under **Additional settings** > **Sign in with an external camera or fingerprint reader**, there's a toggle that allows you to enable or disable ESS:

- When the toggle is **Off**, ESS is enabled and you can't use external peripherals to sign in. Remember, you can still use external peripherals within apps like Teams
- When the toggle is **On**, ESS is disabled and you can use Windows Hello compatible peripherals to sign in

Windows Hello and Companion Device Framework

Article • 11/05/2020

Windows Hello works with the Companion Device Framework to enhance the user authentication experience. Using the Windows Hello companion device framework, a companion device can provide a rich experience for Windows Hello even when biometrics are not available (e.g., if the Windows 10 desktop lacks a camera for face authentication or fingerprint reader device, for example).

> **Note**
>
> The API for the Windows Hello companion device framework is deprecated in Windows 10, version 2004 and later.

Windows Hello biometric requirements

Article • 11/09/2020

Learn about the hardware requirements for biometric equipment, such as IR camera and fingerprint readers in order to support Windows Hello.

Terminology

Term	Definition
False Accept Rate (FAR)	Represents the number of instances that a biometric identification solution verifies an unauthorized person. This is normally represented as a ratio of number of instances in a given population size, for example 1 in 100,000. This can also be represented as a percentage of occurrence, for example, 0.001 percent. This measurement is heavily considered the most important with regards to the security of the biometric algorithm.
True Accept Rate (TAR)	Represents the number of instances a biometric identification solution verifies the authorized user correctly. This is normally represented as a percentage. It is always held that the sum of the True Accept Rate and False Reject Rate is 1.
False Reject Rate (FRR)	Represents the number of instances a biometric identification solution fails to verify an authorized user correctly. Usually represented as a percentage, the sum of the True Accept Rate and False Reject Rate is 1.
Confidence	The confidence in a claimed FAR represents the robustness of the analysis performed in verifying a claimed FAR. Depending on the target or claimed FAR and the importance of the target use case, confidence levels can be varied.
Biometric sample	This refers to the complete biometric sample required to perform a verification operation. For example, a single fingerprint is required to perform a verification.
Biometric spoof	This refers to a synthetic replica of a biometric sample.

Fingerprint reader requirements

Large Area sensors (a sensor matrix of 160 x160 Pixels or more at a dpi of 320 or greater):

- FAR < 0.001%.
- Effective, real world FRR with antispoofing or liveness detection <10%.
- presentation attack defense measures are a requirement.

Small Area sensors (a sensor matrix of less than 160x160 at a dpi of 320 or greater):

- FAR < 0.002%.
- Effective, real world FRR with antispoofing or liveness detection <10%.
- Presentation attack defense measures are a requirement.

Swipe sensors:

- FAR < 0.002%.
- Effective, real world FRR with antispoofing or liveness detection <10%.
- Antispoofing measures are a requirement.

Facial feature recognition requirements

- FAR < 0.001%.
- TAR > 95%.

Appendix

The number of comparisons required to verify a particular level of confidence in a claimed FAR is shown below:

`# of Unique Comparisons = C = 1/((1-Conf)) × 1/((FAR))`

where *FAR* is the desired False Accept Rate and *Conf* is the desired Confidence.

For example, with a desired FAR of 0.001%, at a confidence of 96%:

`# of Unique Comparisons = C = 1/((1-0.96)) × 1/((0.00001))`

`C = 25 × 100,000`

`C = 2,500,000`

In this case, 2,500,000 comparisons would be required to reach the desired confidence in the claimed FAR.

To determine the number of unique biometric samples, *n*, to be collected to achieve these comparisons, the formula below can be used:

`# of Comparisons = n!/2(n-2)!`

`C = n(n-1)/2`

`∴ n^2-n = 2C`

where *n* is the number of unique biometric samples.

In the cases where *n*^2>>*n*, the above formula can be simplified to:

`n^2 ≈ 2C`

`∴ n ≈ √2C`

Continuing with the example above, the number of unique biometric samples needed would be:

`n ≈ √(2×2,500,000)`

`n ≈ 2,236.1`

Meaning about 2,237 unique biometric samples will be needed to verify the confidence in the claimed FAR.

Related topics

Windows Hello face authentication

Windows Hello

Windows Hello: Steps to submit a fingerprint driver

Article • 11/05/2020

Submitting a fingerprint driver for Windows Hello compatibility

Microsoft has introduced new requirements on biometric sensors to comply with Windows Hello quality guidelines. A new manual review process will be necessary to gain approval to interoperate with Windows Hello. The process will be enforced with an OS check for a specific signature obtained through the Windows Partner Center https://partner.microsoft.com/dashboard/home that can only be obtained by undergoing the process in this document. Drivers that have been created and signed by WHQL before 6/1/17 are grandfathered. New and updated drivers that do not obtain this signature after this date will not work with Windows Hello in Window 10, version 1703 or later after the enforcement date.

A driver will always undergo manual approval to obtain the Windows Hello signature. Updates to approved drivers can refer to previous submissions for faster approval. Drivers must undergo a new review if it applies to a new sensor, or if changes to the matching engine have occurred that impact FAR, FRR, or presentation attack detection.

The biometric signature enforcement date is 6/1/2017, after which drivers that do not contain the bio signature will not be loaded and will no longer work with Windows Hello.

Step One: Create a biometric driver

Follow the instructions here to create a biometric driver: Windows Biometric Framework

Step Two: Test your sensor and self-validate

Self validate the sensor and driver to ensure they meet Microsoft's biometric requirements and report findings in the Fingerprint Security Review Template. Documents for the requirements and template can be found within the Fingerprint partner package on Connect. If you do not have access to Connect, contact your Microsoft representative.

Step Three: Modify the driver configuration xml file

When you submit your driver, the Windows 10, version 1703 Fingerprint HLK test will check to ensure that the <vendorCompliance> and <securityReview> tag are included with the following fields:

bugId: ID number for the previous HLK submission that contains the previously approved security review information or 0 if the submission is undergoing an entirely new security review.

updateExistingSubmission: true if the submission serves as an update to a previous submission that has undergone the security review and false if otherwise.

Example

XML

```
<?xml version="1.0" encoding="utf-8"?>
<bioTestConfiguration version="0" runOptional="false" runInteractive="true"
abortOnFailure="false" manualStep="false" priority="3" logType="WTT">
 <vendorCompliance>
   <securityReview bugId="12345678" updateExistingSubmission="true"/>
 </vendorCompliance>
 <testSuites>
   <testSuite deviceRequired="false" id="StorageAdapter">
     <library>storagetest.dll</library>
     <description>storage Adapter Test Suite</description>
   </testSuite>
 </testSuites>
 <deviceInfo>
        <sensorAdapterLib>WinbioSensorAdapter.dll</sensorAdapterLib>
        <engineAdapterLib>vcsWBFEngineAdapter.dll</engineAdapterLib>
        <storageAdapterLib>winbiostorageadapter.dll</storageAdapterLib>
        <indicatorSupported>0</indicatorSupported>
        <supportedModes>
            <supportedMode>0x01</supportedMode>
        </supportedModes>
        <supportedPurposes>
            <supportedPurpose>0x01</supportedPurpose>
            <supportedPurpose>0x02</supportedPurpose>
            <supportedPurpose>0x04</supportedPurpose>
        </supportedPurposes>
 </deviceInfo>
</bioTestConfiguration>
```

Step Four: Modify the driver configuration inf

Biometric driver packages will need to be submitted to the new DevCenter portal to obtain the required Windows Hello signature and be uploaded to WU. Packages will need to include specific properties in the driver INF file to properly specify the adaptor dll's obtaining the digital signature. The following example demonstrates the formatting to obtain the bio signature on adaptor binaries and their related libraries.

For example, if the driver package contained a sensor, engine, and storage adaptor named sensor.dll, engine.dll, and storage.dll respectively, and one loaded stringparser.dll, then to obtain the bio signature on each one, the INF file would have to include the following components:

XML

```
[SignatureAttributes]
sensor.dll = SignatureAttributes.WindowsHello
engine.dll = SignatureAttributes.WindowsHello
storage.dll = SignatureAttributes.WindowsHello
stringparser.dll = SignatureAttributes.WindowsHello

[SignatureAttributes.WindowsHello]
WindowsHello = true
```

This step is the most important to making sure your driver receives the proper certification. All third party biometric adaptor files and any third party dlls loaded by these adaptors will need to be labeled and included in this manner if they are to obtain the biometric signature when submitted to DevCenter.

Step Five: Run the HLK test suite

The HLK tests will make sure the above modifications have been made in steps 3 and 4 and will fail if the configurations information is not there. When packaging the final HLK in HLK studio include the security review template submitted in the bug as a supplemental file.

Step Six: Submit the driver package and HLK logs

Submit the packaged HLK file to DevCenter for review. The feature team within Microsoft will be notified of the submission when it reaches the manual review process. The team will review the submitted template in the HLK package to make sure the self-validated information meets the Microsoft's biometric requirements.

Step Seven: Wait for Microsoft approval and signing

Microsoft will approve the submission provided it meets all Biometric requirements Obtaining the biometric signature is not certification that the driver will work with Windows Hello. For example, a file could be excluded in the inf configuration file that is checked for the signature. If this file is loaded at the time the OS enforces the signature, the load will fail and the driver will not operate with Windows Hello. The signed driver should be tested by the IHV and OEM to ensure it works in the collective system.

Step Eight: Update an existing driver

If an update to a previously signed driver needs to be made, follow the instructions under step 3 for filling in the bugId and updateExistingSubmission fields in the driver configuration xml for the updated driver. If an update is being made to a grandfathered driver, the same steps should be used. The bugId field should be set to the submission ID of the grandfathered driver and the updateExistingSubmission field should be set to true. The driver configuration xml should be included in the driver package that is submitted.

Related topics

Windows Hello face authentication

Windows Hello

Windows Hello face authentication

Article • 07/15/2021

Microsoft face authentication in Windows 10 is an enterprise-grade identity verification mechanism that's integrated into the Windows Biometric Framework (WBF) as a core Microsoft Windows component called Windows Hello. Windows Hello face authentication utilizes a camera specially configured for near infrared (IR) imaging to authenticate and unlock Windows devices as well as unlock your Microsoft Passport.

Key benefits and capabilities of Windows Hello face authentication

These are the key benefits to using the Windows Hello face authentication:

- Facial recognition across all Windows 10-based devices and platforms with compatible hardware (near IR sensor).
- A user-friendly interface that provides single sign form of verification to unlock your Microsoft Passport.
- Enterprise-grade authentication and access to Microsoft Passport Pro supported content, including network resources, websites, and payment instruments.
- The ability to provide a consistent image (using IR) in diverse lighting conditions that also allows for subtle changes in appearance including facial hair, cosmetic makeup, and so on.

Scenarios

The two primary scenarios for Windows Hello face authentication in Windows 10 are authentication to log on or unlock, and re-authentication to prove you are still there.

Authentication

Expand table

Type	Description
average duration	< 2 seconds
expected frequency	High
frequency	Occurs every time a user wants to unlock their device or moves past the

Type	Description
description	lock screen

Re-authentication

Expand table

Type	Description
average duration	< 2 seconds
expected frequency	Low
frequency description	Occurs when an application or web site would like to re-verify that the user is in front of their device

How it works

The Windows Hello face recognition engine consists of four distinct steps that allow Windows to understand who is in front of the sensor:

1. **Find the face and discover landmarks**

   In this first step, the algorithm detects the user's face in the camera stream and then locates facial landmark points (also known as alignment points), which correspond to eyes, nose, mouth, and so on.

2. **Head orientation**

   To ensure the algorithm has enough of your face in view to make an authentication decision, it ensures the user is facing towards the device +/- 15 degrees.

3. **Representation vector**

   Using the landmark locations as anchor points, the algorithm takes thousands of samples from different areas of the face to build a representation. The representation at its most basic form is a histogram representing the light and dark differences around specifics points. No image of the face is ever stored – it is only the representation.

4. **Decision engine**

Once there is a representation of the user in front of the sensor, it is compared to the enrolled users on the physical device. The representation must cross a machine-learned threshold before the algorithm will accept it as a correct match. If there are multiple users enrolled on the system, this threshold will increase accordingly to help ensure that security is not compromised.

Enrollment

Enrollment is the step of generating a representation or set of representations of yourself (for example if you have glasses you may need to enroll with them and without them) and storing them in the system for future comparison. This collection of representations is called your enrollment profile. Microsoft never stores an actual image and your enrollment data is never sent to websites or applications for authentication.

Most users will likely need to enroll once per device. Additional enrollments are needed for users that:

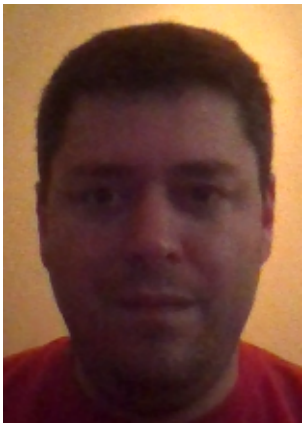
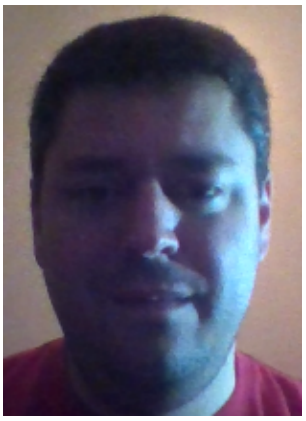
- Occasionally wear certain types of glasses
- Have had major changes to facial shape or texture
- Move to environments with high ambient near IR light (for example, if you take your device outside in the sunshine)

Benefits of near infrared

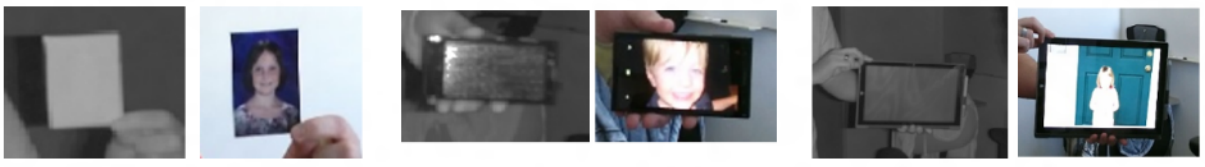
After the release of face recognition with the first Kinect on Xbox 360, Microsoft learned that relying on ambient light to provide a consistent image provided a poor user experience. People live and work in a variety of environments, with an assortment of lighting conditions. Traditional color recognition systems rely on turning up the brightness, exposure, or other settings to create a useable image – all of which expose artifacts that impact the robustness of the system.

In contrast, near infrared images are consistent across ambient lighting scenarios, as you can see below.

Expand table

Scenario	Color Image from integrated Camera	IR Image from Microsoft Reference Sensor
Low light representative of watching TV or giving a PowerPoint presentation		
Side lighting when sitting near a window or desk lamp		

Using IR also helps with spoofing because it helps prevent the most accessible attacks. For instance, IR doesn't display in photos because it's a different wavelength, and as you can see below, the images the images do not display in photos or on an LCD display.



How accuracy is measured

When Microsoft talks about the accuracy of Windows Hello face authentication, there are three primary measures used: False Positives, True Positives, and False Negatives.

Expand table

Term	False Positive	True Positive	False Negative
Description	Sometimes also calculated as a False Acceptance Rate, this	The True Positive rate represents the likelihood a	Represents the likelihood a user is

Term	False Positive	True Positive	False Negative
	represents the likelihood a random user who obtains physical access to your device will be recognized as you. This number should be as low as possible.	user will be correctly matched to their enrolled profile each time they are positioned in front of the sensor. This number should be high.	not matched to their enrolled profile. This number should be low.
Windows 10 Algorithm	Less than 0.001% or 1/100,000 FAR	Greater than 95% with a single enrolled user	Less than 5% with a single enrolled user

Accounting for errors in measurement is important, so Microsoft categorizes them in two ways: bias errors (systematic errors) and random errors (sampling).

Bias errors

Bias errors may occur as a result of not using data that is representative of the environments and the conditions in which the algorithm is used. This type of error can result from different environmental conditions (such as lighting, angle to sensor, distance, and so on) as well as hardware that is not representative if shipping devices.

Random errors

Random errors results from using data that doesn't match the population diversity that will actually be using the feature. For example, focusing on a small set of faces without glasses, beards, or unique facial features.

External camera security

It's strongly encouraged to run Windows Update constantly and make sure your system is updated with the latest security updates, including the updates released on July 13, 2021 to improve security when using Windows Hello camera described in CVE-2021-34466. In addition, if you want to disallow the use of external Hello camera completely, you can add an optional registry value in the following path.
Registry path:
`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Authentication\LogonUI\FaceLogon`
DWORD value: `ShouldForbidExternalCameras`
Value: `1`

Related topics

Windows Biometric Framework API

Windows Hello fingerprint authentication

Article • 11/09/2020

Starting with Windows 10, version 19H1, Windows supports a new enrollment experience for fingerprint. This change provides an intuitive experience that makes enrollment easier for users. Windows 10 provides this new experience regardless of device type or sensor.

The new experience consumes information about a sensor to provide enhanced instructions. This information must be provided by an OEM or IHV in the form of registry keys that can be written by the sensor driver.

Sensor information for enhanced instructions

Windows uses the fingerprint sensor information to identify whether to load specific graphical and textual instructions during the fingerprint enrollment experience. The information that manufacturers can provide is:

- The location of the sensor on the device.
- The shape of the sensor.
- Whether the sensor is also mapped to another button.

Windows may not consume all of the information provided in the first implementation.

Format of Sensor Information

Registry path

The information about a fingerprint sensor must be written to the following registry path by the sensor driver package:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services**WbioSrvc\Service Providers\Fingerprint\Sensor Layout\<Hardware ID for sensor>**

> ⓘ **Note**
>
> The Hardware ID must be scrubbed to remove any backslashes before it can be used as a registry key name. Any backslashes should be replaced with a single

blank space.

Registry values

The actual sensor information will be written as REG_DWORDs to the path shown above. These registry values will be decimals that correspond to specific sensor characteristics. The names of the registry values that can be provided are the following:

- location
- shape
- powerbutton
- startbutton

The following values and their mappings are defined for these categories:

location

Expand table

Value	Mapping
0	Not Specified
1	Device front
2	Device back
3	Left side
4	Right side
5	Device Top
6	External Sensor
7	Dual screen right
8	Keyboard Top Row Right First Key
9	Keyboard Top Row Right Second Key
10	Keyboard Bottom Row Any Key
11	Keyboard Unspecified Key

shape

Expand table

Value	Mapping
0	Not Specified
1	Square
2	Rectangle
3	Circle
4	Oval

powerbutton

Expand table

Value	Mapping
0	No
1	Yes

startbutton

Expand table

Value	Mapping
0	No
1	Yes

Default Values

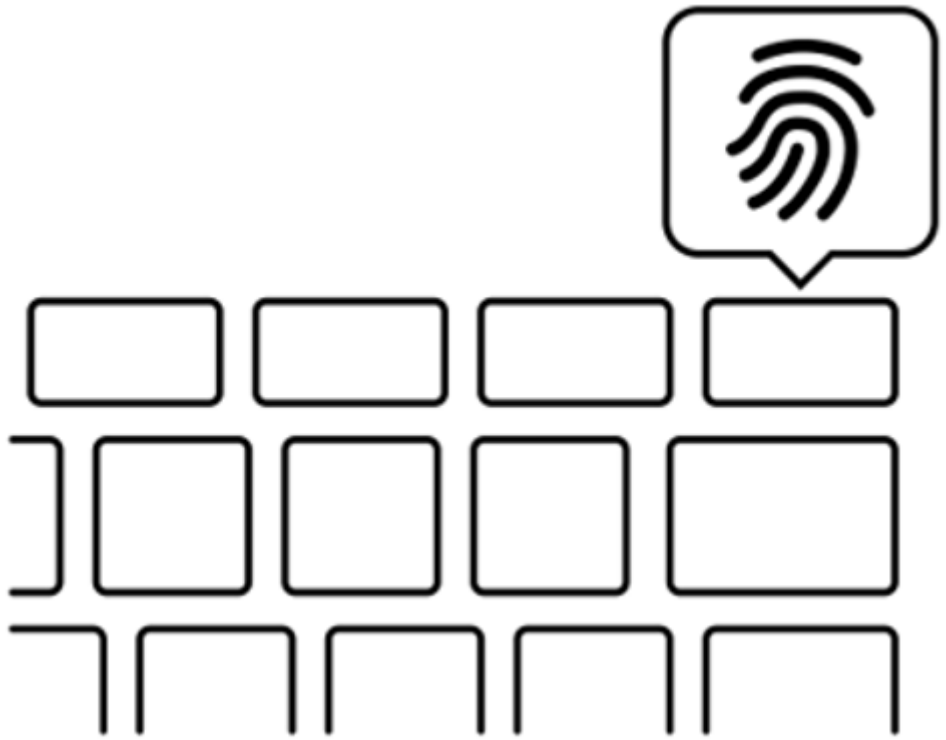
- Location: Not Specified (0)
- Shape: Not Specified (0)
- Power Button: No (0)
- Start Button: No (0)

Sample keyboard layouts

The following images illustrate the use of the registry values to models of keyboards.

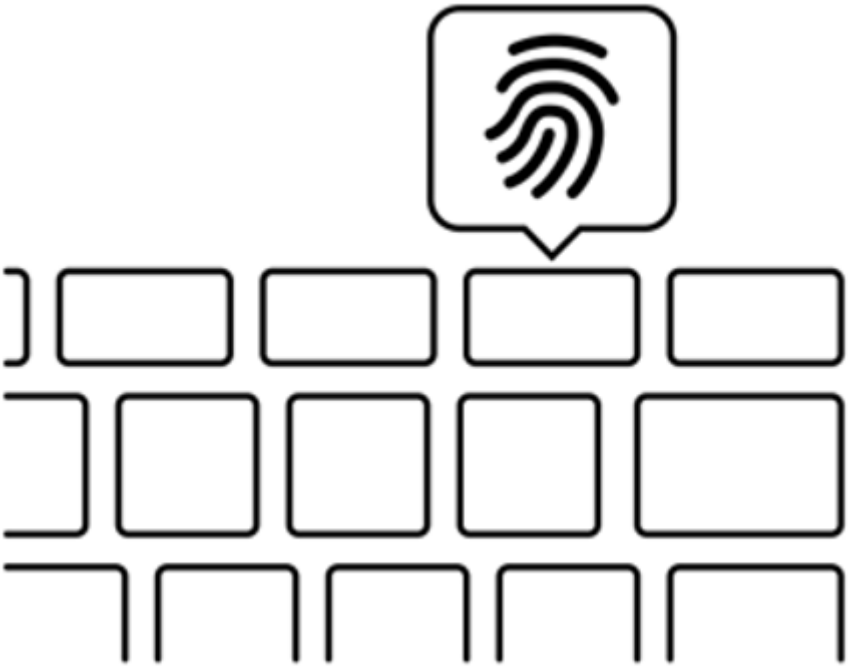
Sensor on the Top Right Keyboard Power Button

OEM sets Location: 8 and PowerButton: 1



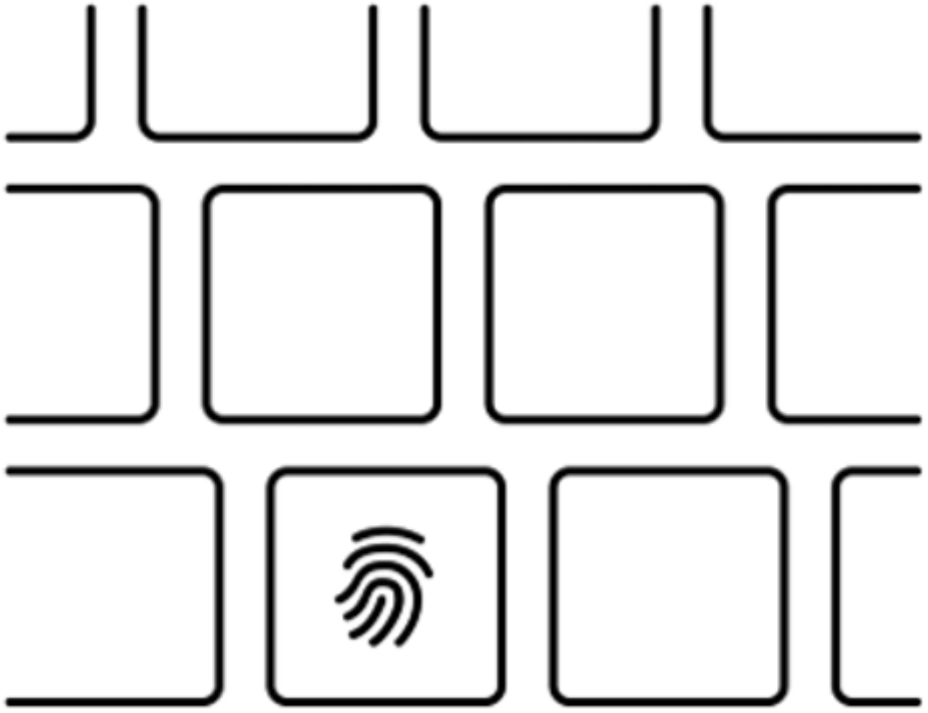
Sensor on the Second from the Right Keyboard Power Button

OEM sets Location: 9 and PowerButton: 1



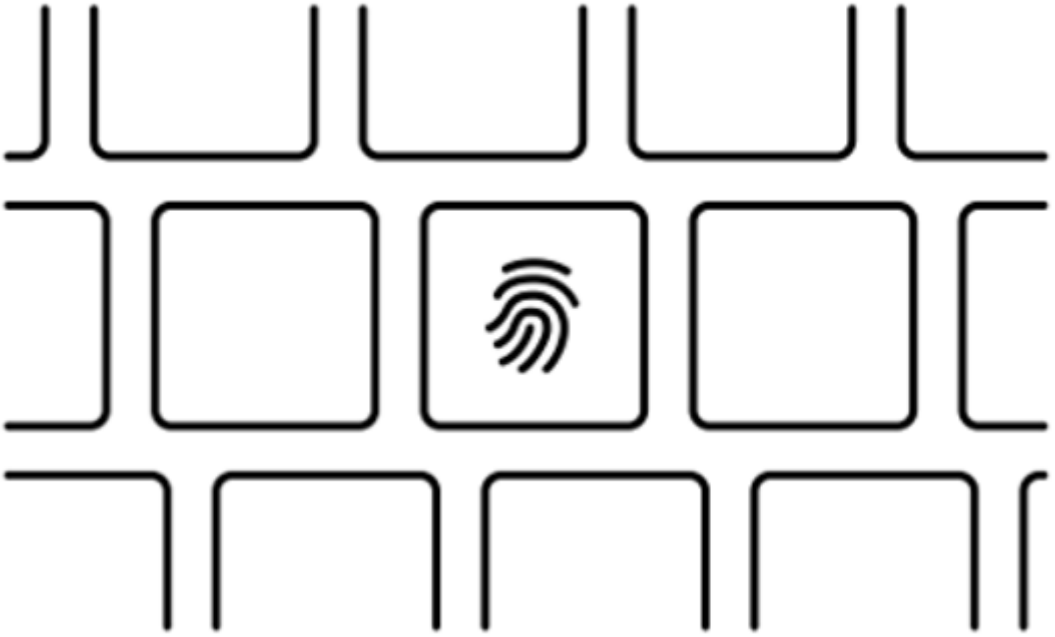
Sensor on the Keyboard Bottom Row

OEM sets Location: 10



Sensor Generically Located on Keyboard

OEM sets Location: 11



Example

The following registry settings describes a circular fingerprint sensor on the front of the device, which is neither a power button or home button:

C++

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\WbioSrvc\Service
Providers\Fingerprint\Sensor Layout\USB VID_0000&PID_0000]
"location"=dword:00000001
"shape"=dword:00000003
"powerbutton"=dword:00000000
"startbutton"=dword:00000000
```

What is Modern Standby

Article • 03/03/2021

Windows 10 Modern Standby (Modern Standby) expands the Windows 8.1 Connected Standby power model. Connected Standby, and consequently Modern Standby, enable an instant on / instant off user experience, similar to smartphone power models. Just like the phone, the S0 low power idle model enables the system to stay connected to the network while in a low power mode.

Although Modern Standby enables an instant on/off user experience like Connected Standby, Modern Standby is more inclusive than the Windows 8.1 Connected Standby power model. Modern Standby allows for market segments previously limited to the Traditional Sleep (S3) power model to take advantage of the low power idle model. Example systems include systems based on rotational media and hybrid media (for example, SSD + HDD or SSHD) and/or a NIC that doesn't support all of the prior requirements for Connected Standby.

The number of systems supporting Modern Standby rather than S3 is increasing over time. The Modern Standby section outlines important changes, partner requirements, and best practices for enabling Modern Standby.

> **Note**
>
> Modern Standby is available for both Windows 10 desktop and Windows 10X.
>
> Switching between S3 and Modern Standby cannot be done by changing a setting in the BIOS. Switching the power model is not supported in Windows without a complete OS re-install.

Functional Overview of Modern Standby

A Modern Standby session encompasses the entire screen-off to screen-on user scenario. However, individual *screen off* and *sleep* segments make up an overall Modern Standby session. These states differentiate periods of quiescing software behavior from periods of long term low power. For this reason, Microsoft conceptualizes modern sleep as equivalent to traditional S3 sleep, with the added benefit of allowing value-added software activities to run periodically.

In Windows 10, low power is achieved by only waking from the lowest power state when absolutely necessary and only allowing software to execute in short, controlled bursts of

activity, dramatically reducing the opportunities for software components to execute. Windows and the SoC hardware are always listening for interesting events (such as a network packet or user input at a keyboard) and will wake up instantly when needed. The system will wake when there is real time action required, such as for OS maintenance or when a user wakes the system.

Modern Standby consists of multiple hardware and software power modes, all of which occur with the screen turned off. The complexity of Modern Standby is a result of keeping the system alive to process background tasks, while ensuring that the system stays quiet enough to achieve long battery life.

Entry to Modern Standby

Modern Standby starts when the user causes the system to enter sleep (e.g user pressing the power button, closing the lid, idling out, or selecting Sleep from the power button in the Windows Start menu). On entry to Modern Standby, apps and system software must be made ready for the transition to low-power operation. (See Prepare software for Modern Standby.) After software components and apps have been prepared for low-power operation, hardware components, including their software device drivers, must be similarly prepared for low-power operation. (See Prepare hardware for Modern Standby.) **Both software and hardware must be made ready for low-power operation.**

Activity during Modern Standby

On-demand transitions to active mode can occur in response to user inputs, interrupts from networking devices and other hardware events. Windows transitions the SoC from active mode to idle mode after all software activity is stopped and the devices on and off the SoC have entered low-power states. (See Transitioning between active and idle states.)

The networking and communications devices automatically transition between active and low-power modes based on the software activity of the system during Modern Standby. When there are no system services or Microsoft Store app background tasks that require the network, the networking device is in the low-power, protocol offload, and WoL patterns mode. When a system service or background task requires network access, Windows automatically transitions the networking device to an active mode.

On occasion, the system stays in the active mode (with the screen off) for a longer interval of time. These longer active intervals occur for a variety of reasons, for example, processing incoming email or downloading critical Windows updates. Windows

components that are allowed to leave the SoC in the active power state are called activators because they are registered with the power manager as capable of blocking the transition back to the idle power mode. The durations of these activities vary widely but are controlled to extend battery life. The durations of the activities can be viewed with the built-in SleepStudy software tool or through Event Tracing for Windows (ETW)-based instrumentation.

On Windows 8.1 Connected Standby systems, during Modern Standby, Windows transitions the SoC from idle mode to active mode a minimum of every 30 seconds to perform kernel maintenance tasks. This maintenance activity is extremely brief in duration (typically no more than a few hundred milliseconds) and cannot be adjusted. This does not happen on Windows 10 Modern Standby systems.

Resume from Modern Standby

When the user causes the system to resume from standby, e.g. presses the power button, the display is immediately turned on and networking devices are restored to their normal, active operating modes. The time from the power button press to the display turning on is less than one second. After the display is turned on and the networking device returns to normal operating mode, desktop applications resume and the system returns to its normal, screen-on active behavior.

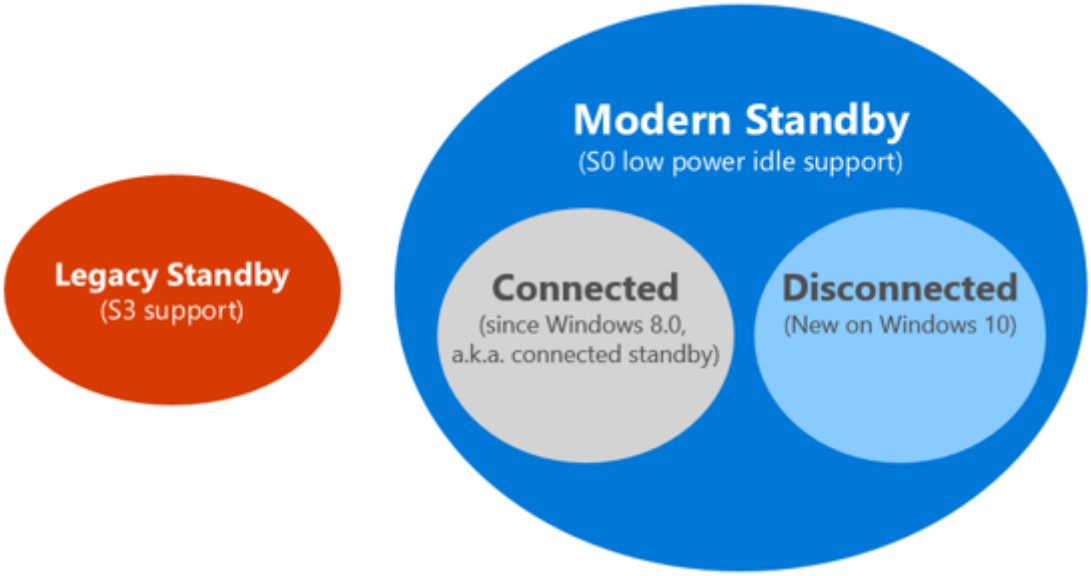
Related Articles

- Modern Standby vs S3
- Modern Standby key concepts overview
- Modern Standby states
- Prepare hardware for modern standby
- Prepare software for modern standby
- Activators
- SleepStudy

Modern Standby vs S3

Article • 11/05/2020

In Windows 10, there are two power models for PCs: S3 and Modern Standby. The S3 power model is an older standard and is not capable of the instant on that consumers expect from modern devices. Modern Standby is capable of leveraging all the capabilities of a modern chipset and can be integrated across the breadth of tablets and PCs today. The first iteration of Modern Standby was Connected Standby, which first shipped in Windows 8 and Windows 8.1. Modern Standby expands upon the Windows 8.x Connected Standby concept, allowing for flexibility in component selection and the ability for the OS to manage network connectivity in standby.



The above figure illustrates the relationships between the models and how Modern Standby systems can either stay connected to or disconnect from the network while in standby.

On any Modern Standby system, the system remains in S0 while in standby, allowing the following scenarios to work:

- Background activity
- Faster resume from a low power state

On systems that can stay connected while in standby, wakes based on specific network patterns may also be set by the operating system to enable apps to receive the latest content such as incoming email, VoIP calls, or news articles.

Conceptualization

Starting in Windows version 2004, Microsoft has defined modern standby states to better align with the S3 power model. Modern Standby comprises *Screen Off* and *Sleep* states, where the *Screen Off* state encompasses active behaviors to quiesce the system to sleep, and the *Sleep* state is equivalent to S3 sleep with the added benefit that explicitly-allowed, value-adding software activity can run.

Technical differences

When in the lowest power state, systems may look very similar to systems in the S3 state —processors are powered off and memory is in self-refresh. The difference is in the path of how it enters and exits low power state. For S3 systems, the system is either active or in S3. For Modern Standby, the transition from the active to the low power state is a series of steps to lower power consumption. Components are powered down when they are not in use. So, the transition into and out of a lower power state is much quicker on a Modern Standby system than on an S3 system. This design also helps with the speed of entry and exit from Standby as it doesn't require firmware interactions.

AC power vs. battery power

Modern Standby systems can go into maintenance mode while on AC power. If a maintenance task occurs while on AC power, updates and other activities could occur and use the network if a connection is available.

Advantages of using Modern Standby over S3

Instant On

An instant on experience is a key value available in Modern Standby. Telemetry data indicates that resume time from the low power idle model is at least twice as fast as that from an S3 resume.

Users expect their PC and tablet devices to instantly power on, the same way their smartphones do. The time it takes for PCs and tablets to power on is greatly reducing and Windows devices need to have near instant-on capabilities.

Background activity while the system is "off"

When Modern Standby-capable systems enter sleep, the system is still in S0 (a fully running state, ready and able to do work). Desktop apps are stopped by the Desktop

Activity Moderator (DAM); however, background tasks from Microsoft Store apps are permitted to do work. In connected standby, the network is still active, and users can receive events such as VoIP calls in a Windows store app. While VoIP calls coming in over Wi-Fi wouldn't be available in disconnected standby, real-time events such as reminders or a Bluetooth device syncing can still happen.

Simplified wake story

Device wake, even in the S3 model, requires functionality integrated into the BIOS. This makes wake times slower and more cumbersome to develop. With Modern Standby a system stays in the S0 state and waking simply requires a hardware interrupt. Thus, it may eliminate any need for firmware interaction.

Windows 8.1 Connected Standby systems wake up the SoC at least once every 30 seconds to process work.

Windows 10 is optimized to deliver longer battery life by postponing non-critical work and removing unnecessary wakeups during Modern Standby, particularly when the system is operating on battery power. Device interrupts continue to operate as normal to allow incoming instant messages, notifications, phone calls, etc. Some timers are postponed indefinitely until the system is awake or plugged into AC power to reduce the amount of power consumed by processing non-critical work when the system is "asleep".

Modern Standby systems have the same wake capabilities as Windows 8.1 Connected Standby systems based on events. Wake capabilities make it easier to support a wider range of wake scenarios, such as wake-on-Bluetooth device events.

The use of legacy wake events is possible; however, it would require the ability to arm a device for wake when going into a D3 state. For example, a wake on LAN would be expected to work the same way on a Modern Standby system and an S3 system.

For more information on wake support, see Modern standby wake sources.

Key concepts overview

Article • 11/05/2020

Modern standby achieves low power and constant connectivity by spending the majority of time in a low-power state in which the SoC is powered down (DRIPS).

After the display is powered off, Windows transitions through a set of phases to prepare apps and system software for achieving low power. The purpose of these phases is to stop or reduce as much software activity on the system as possible. After software components and apps have been prepared for low-power operation, hardware components, including their software device drivers, must be similarly prepared for low-power operation.

Windows only powers up the SoC when activators block the transition back to the idle power mode to perform scheduled background activity or to respond to incoming network activity or device interrupts. Windows achieves low power consumption by aggressively entering the idle power mode and controlling the amount of activity.

If devices block software DRIPS (SW DRIPS) for a certain length of time in the absence of any activator-brokered software activity, the OS can direct devices to enter their appropriate low-power idle states through Directed Deepest Runtime Idle Platform State (DDRIPS) mechanisms.

Modern standby systems can enter connected standby or disconnected standby on a per-Modern Standby session basis based on user opt-in to connected scenarios, described in more detail in Network connectivity.

If the system is in disconnected standby, very little activity occurs during sleep. The system should spend most of its time in the idle power mode (DRIPS).

If the system is in connected standby, the amount of activity is typically higher. This activity is mostly the result of app background tasks that require a network connection to be usable. For example, email will not sync without an Internet connection. When the system is in connected standby and an active Internet connection is present, the system will transition between the idle and active modes on demand if incoming network packets match Wake on LAN (WoL) patterns.

Modern Standby States

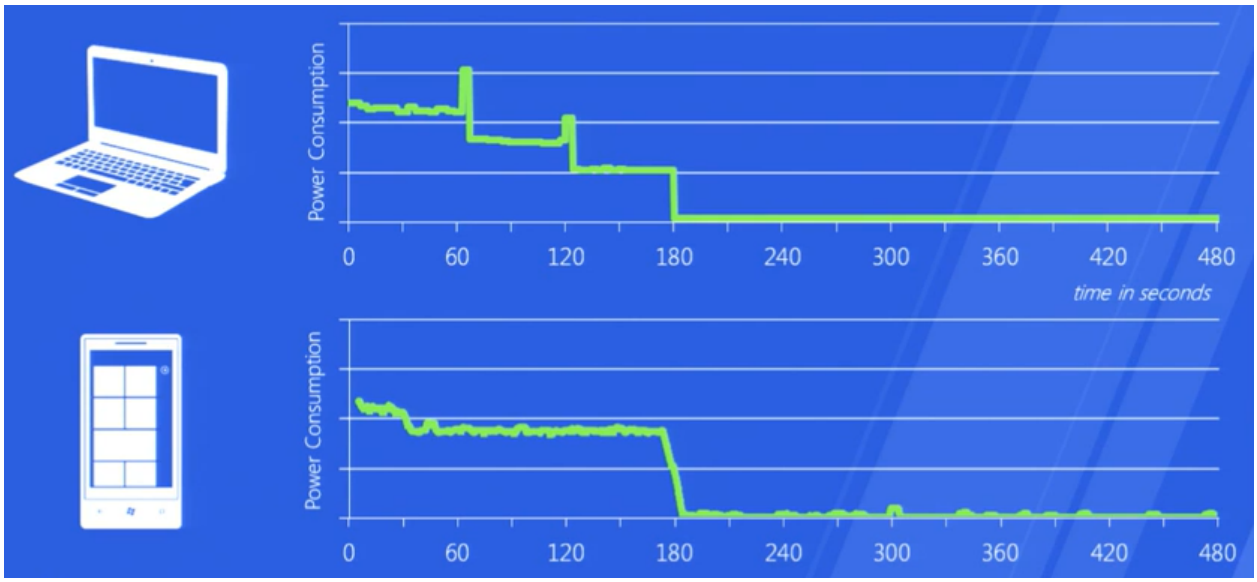
Article • 03/03/2021

From a user's perspective, a modern standby "session" is conceptualized as the time between two system events: screen off to screen on. However, within this overall session, several layers of system activities will run as the system is quiesced to enter its lowest power state. Because of this, Microsoft conceptualizes the screen off to screen on modern standby session as two separate states: *Screen Off* and *Sleep*. This article aims to explain the distinction of these states. Please see Modern Standby firmware notifications to learn about the transition notifications available to system designers.

Modern Standby system states

There is a clear distinction between software activity that can run as the system is being quiesced and software activity that can run while the system is in its true sleep phase. Thus, the expected behaviors and performance standards differ between the *Screen Off* state and the *Sleep* state.

After the screen turns off, a waterfall flow of phases quiets the system to its Sleep state. Please see this page on preparing the software for modern standby for more detailed information on this process. When the system reaches the "low power phase", all ancillary, larger workloads have completed and it is able to enter into a *Sleep* state. During the *Sleep* state, specific value-adding SW activity may run— e.g., the system receives an email notification, refreshes a network connection, etc.—while maintaining a very low power floor. Thus, like a smartphone, a modern standby system aims to consume very low energy while still providing the user real-time connectivity and speedy wake times. While in *Sleep*, the system primarily remains in DRIPS--where no software activity runs whatsoever-- yet it may wake briefly to perform value-adding SW activities.



A generic illustration of the typical power model (top) versus the smartphone-like Modern Standby power model (bottom). In the traditional S3 model, the system performs a few main tasks as it is quiesced to S3 sleep, in which no software activity runs at all. In the Modern Standby power model, the system is gradually quiesced to sleep, where small bursts of well-managed, value-adding software activity will run.

Beginning in Windows 10 version 2004, the differentiation between *Screen Off* and *Sleep* is also reflected in the SleepStudy report ('powercfg spr') as shown below. The top-level heading will indicate whether the system is Modern Standby-enabled or following the Traditional Sleep (S3) power model. On Modern Standby-capable systems, the session summary breakout will separate by *Screen Off* and *Sleep*.

States:
◉ All ○ Active ○ Screen Off ○ Sleep ○ Hibernate ○ Shutdown ○ Bug Check

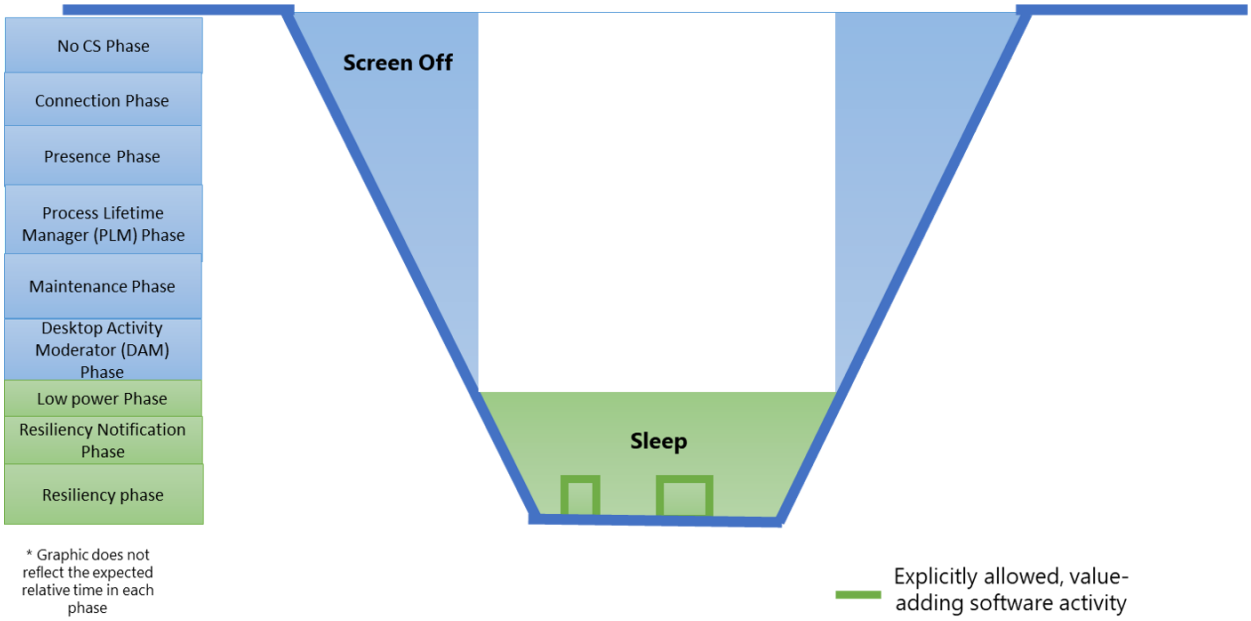
	START TIME	DURATION	STATE	ENERGY CHANGE		CHANGE RATE		% LOW POWER STATE TIME		% CAPACITY REMAINING AT START
1	2019-05-02 16:46:20	00:02:13	Screen Off	612 mWh	0% of battery	204 mW	Drain	-		100%
2	16:48:33	3:00:14	Sleep	3,792 mWh	6% of battery	1,263 mW	Drain	SW: 82%	HW: 63%	100%
3	19:48:13	89:38:06	Hibernate	-	-	-	-	-		-
4	2019-05-06 13:26:54	0:01:44	Active	473 mWh	1% of battery	16,373 mW	Drain	-		94%
5	15:59:13	0:00:31	Screen Off	51 mWh	0% of battery	153 mW	Drain	-		93%
6	15:59:44	0:02:57	Sleep	268 mWh	0% of battery	5,600 mW	Drain	SW: 100%	HW: 100%	93%
7	16:20:10	00:04:11	Screen Off	36 mWh	1% of battery	843 mW	Drain	-		93%
8	16:48:33	2:04:17	Sleep	1,874 mWh	3% of battery	978 mW	Drain	SW: 96%	HW: 97%	92%
9	18:52:37	0:27:10	Active	3,435 mWh	5% of battery	7,586 mW	Drain	-		89%
10	19:27:01	0:00:00	Report Generated	-	-	-	-	-		-

For more information on how to interpret sleep study reports, please see the Sleepstudy report page.

Summary of key points

- The *Screen Off* state encompasses the period of a Modern Standby session where quiescing behavior and larger workloads are running. The *Sleep* state encompasses the period of a Modern Standby session when the system is in a long-term, low power state and only explicitly allowed, value-adding activities are allowed to run.

- The *Screen Off* state is typically expected to have a short duration. Please see prepare software for modern standby for more details on behaviors that can extend a *Screen Off* segment.
- A Modern Standby session can contain multiple *Screen Off* and *Sleep* segments as the system transitions into long term sleep and wakes for input or larger maintenance workloads.
- The "Low power phase" described in prepare software for modern standby marks the beginning of the *Sleep* phase.



Prepare software for modern standby

Article • 12/22/2021

When a system enters modern standby--that is, when the screen turns off--its software and applications must be made ready to enter its long term, low power sleep state. Please see Modern Standby states for a higher level overview of this behavior.

A system enters modern standby when the display turns off. This will occur in response to:

- the user pressing the power button
- the user closing the lid
- the user selecting *Sleep* from the power button in *Settings*
- the system idling out

Software preparation phases

Following the display turning off, the system will begin quiescing software activities to enter modern sleep. Microsoft defines modern sleep as entering the *low power phase* indicated below. Within modern sleep, explicitly allowed, value-adding software activities can run intermittently, but the system will aim to spend most of the time in its lowest power state, known as *deepest idle runtime platform state (DRIPS)*.

Windows transitions through a set of phases to prepare apps and system software for achieving low power. The phases are sequential—each phase is a set of activities that must be performed before the phase can be completed and the next phase entered.

The purpose of these phases is to stop or reduce as much software activity on the system as possible. This reduction in activity gives more opportunity for the hardware to enter low-power modes, thereby extending overall battery life.

The following table describes each phase of preparing software for modern sleep.

Expand table

Topics	Description	Tasks performed	Exited when...	Typical duration (seconds)
No-CS phase **Note** This is	The system is waiting either to idle out or for outstanding	• The system waits for the sleep timeout to expire.	All outstanding power requests have been cleared	• Typically, less than one second.

Topics	Description	Tasks performed	Exited when...	Typical duration (seconds)
also the phase where the device waits for the sleep timeout to elapse and power requests to expire.	power requests from applications. It has not yet begun quiescing activities.	• The system waits for power requests to expire. These can be held by applications to keep the system active. Power requests will block the NoCS phase indefinitely on AC power, and for up to 5 minutes on DC power. Audio and mobile hotspots are allowed to run indefinitely.	by applications or the maximum time-out has been reached.	• If the system is on battery power, outstanding power requests will cause this phase to block for a maximum of five minutes. The applications with power requests can be inspected by running Powercfg.exe with the /requests option. • If the system is on AC power, outstanding power requests will cause this phase to block indefinitely or until the power request is cleared by the application.
Connection phase	The system is checking for remote desktop connections.	• Determine if remote desktop session(s) exist. • Begin tracking outstanding power requests.	There are no remote desktop sessions connected.	• Zero seconds if no remote desktop sessions are connected. • Phase will last until all

Topics	Description	Tasks performed	Exited when...	Typical duration (seconds)
				remote desktop sessions are disconnected or have timed out.
Process Lifetime Manager (PLM) phase	The system suspends Microsoft Store apps that are in the foreground.	• Suspend all foreground Microsoft Store apps. • Check for communications app activity.	All foreground Microsoft Store apps have been suspended and no audio playback is occurring.	• Typically, less than five seconds.
Maintenance phase	The system executes maintenance tasks.	Wait for maintenance tasks to complete if running (most common on AC power).	No system maintenance tasks are running.	• Typically, less than one second on DC power. • The system is most likely to block on maintenance phase on AC power.
Desktop Activity Moderator (DAM) phase	The system pauses desktop applications to reduce their power consumption during standby.	The system suspends desktop applications	All desktop applications have been suspended and no audio playback is occurring.	Typically, less than one second.
Low-power phase **Note** This is where	The system notifies registered subscribers that the power manager is entering a low-power, long-	Notify registered subscribers.	All registered subscribers have been notified.	Typically, five seconds.

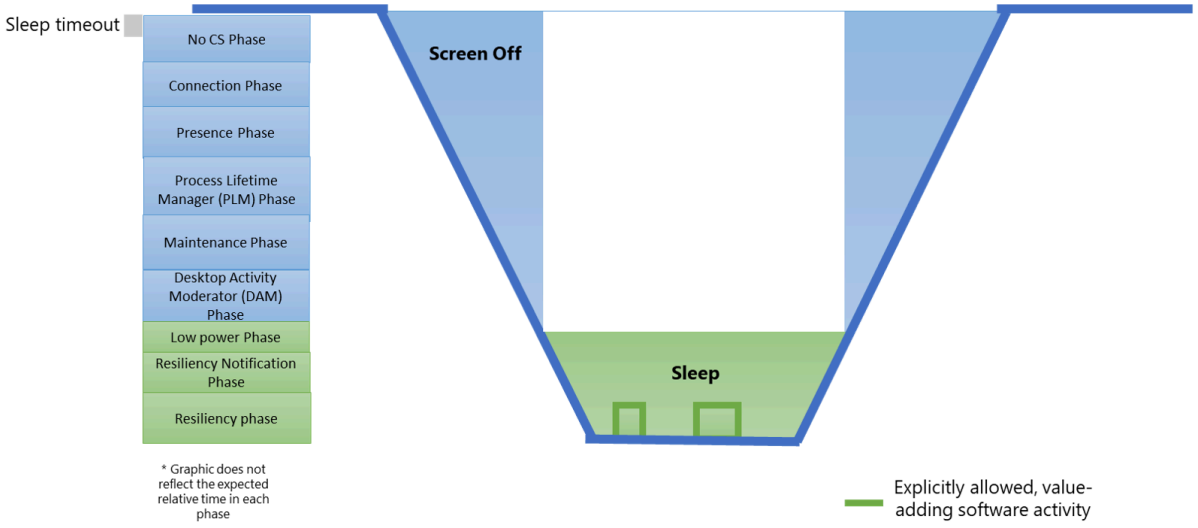
Topics	Description	Tasks performed	Exited when...	Typical duration (seconds)
the system conceptually exits its *Screen Off* state and enters *Sleep*.	resume-latency phase. This is used by some devices as a hint to power down.			
Resiliency notification phase	The network subsystem is notified to enter a low-power mode.	Notify the network subsystem. Network adapters that do not support modern connected standby are turned off (D3).	The network subsystem has been notified.	Typically, less than one second.
Resiliency phase **Note** This is where explicitly allowed, value-adding software activities can run through **activators**. Consequently, the	The system is ready for the SoC to enter the lowest power mode and remain idle.	PDC resiliency clients are notified that the system is in resiliency phase. Session-0 services are throttled by the DAM to no more than one second of activity every 30 seconds. The power manager waits for activators to turn on their reference and cause the system to remain active. As of 24H2, additional session-0 services may be suspended, and session-0 service throttling may be stopped to decrease battery drain.	The system exits standby due to user input or a power button press. The system transitions to maintenance phase to run system maintenance (this would show as a *Screen Off* segment in Sleepstudy).	The majority of time the system is spent in DRIPS.

Topics	Description	Tasks performed	Exited when...	Typical duration (seconds)
system may transition between idle and active states periodically.				

After Windows has entered the resiliency phase, software is considered to be fully prepared for low-power operation. As soon as all devices have entered their low-power modes, the SoC and core chipset will enter the lowest idle power mode, as described in Prepare Hardware for Modern Standby.

Note Currently, no means is available to system designers to change or influence the behavior of these phases—they are a core part of Windows behavior. The details are explained here to aid in system debugging and development.

The following graphic depicts how these phases fit into the overall *Modern Standby* session flow:



Desktop activity moderator

The desktop activity moderator (DAM) is the Windows component that is used to pause all desktop applications and throttle the runtime of third-party system services. The purpose of the DAM is to enable basic software compatibility with existing applications and services, but mitigate their impact on battery life during standby.

Windows prevents desktop applications from running during any part of modern standby after completing the DAM phase. Windows allows third-party system services to execute in a throttled mode after completing the DAM phase.

Prepare hardware for modern standby

Article • 11/05/2020

On entry to modern standby, hardware components must be prepared to make the transition to low-power operation. After software components and apps have been prepared for low-power operation, hardware components, including their software device drivers, must be similarly prepared for low-power operation.

The remainder of this article explains how to prepare the devices outside and inside the System on a Chip (SoC) to operate in a low-power mode after the hardware platform enters standby.

Hardware transition to low-power mode

All devices outside the SoC and inside the SoC must enter a low-power mode in order to achieve long battery life during sleep. After a hardware platform enters sleep, the devices in the platform switch to low-power modes in an orderly process that starts with the devices outside the SoC.

First, all devices outside of the SoC or core silicon must enter a low-power mode. The power mode may be a clock-gated idle state—for example, placing an I^2C-attached touch controller in a sleep mode. Or, the power mode may be a power-gated, 0-watt state called *D3cold*. A USB-attached web camera will often transition to D3cold during modern standby. For more information, see Supporting D3cold for USB Devices.

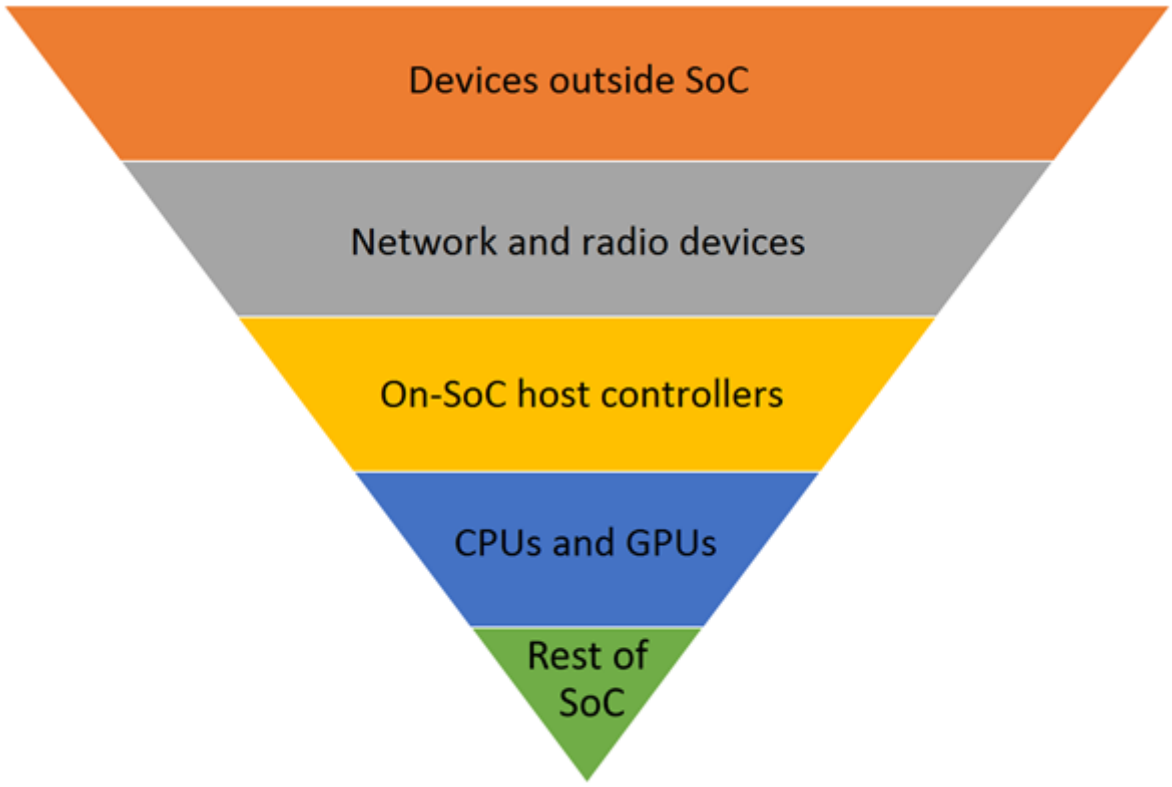
Each device class and connection bus has its own terminology and requirements for transitioning a device to the lowest power mode. However, it is critical that a system designer plan a low-power mode of operation for each device in the platform during modern standby. The battery life of the system and the ability to place the SoC itself in a low-power mode depend on correct power management of each device outside of the SoC itself.

Next, network and radio devices are placed in a low-power mode for sleep. During sleep, these devices often still have power applied to maintain connectivity and are required to wake the SoC if necessary. Communications and radio devices typically enter the D2/D3 low-power state, although the entry into each state is device class-specific and bus-specific.

After all devices outside of the SoC, including communications devices, have been powered down, the host controllers on the SoC will be turned off. Almost every SoC has

USB, I^2C, GPIO, SDIO, and UART host controllers. Each of these components on the SoC must be turned off in order for the SoC to enter a low-power mode.

The process for preparing the hardware for low-power during sleep can be visualized as an upside-down pyramid, as shown in the following diagram. The lowest power is achieved when the whole SoC chip is powered down, but this can occur only after each set of devices above it in the pyramid has been powered down.



Powering down the devices outside the SoC

Each device outside of the SoC chip must enter a low-power mode for two key goals:

- Reduce the power consumption of the device.
- Allow the SoC itself to power down by allowing the on-SoC host controller, to which the device is connected, to power down.

The method to power down each device outside of the SoC varies by device class and attached bus.

Some devices outside of the SoC are placed in a 0-watt, no-power-consumed state called *D3cold*. Common devices for D3cold include cameras and sensors. The driver must save the register state of the device and then transition the device to the D3 power state. Power will be removed by the ACPI firmware by toggling a GPIO line or turning off a power rail from the power management IC (PMIC).

Some devices outside of the SoC are placed in a low-power idle mode in which register state is maintained, or the device might simply be clock-gated. For example, many touch controllers feature a clock-gated state that consumes less than 1 milliwatt of power. The typical advantages to using a clock-gated state are faster power-on time and lower cost by not having to connect the device to a switchable power rail.

Generally, every device outside of the SoC must be capable of entering a low-power mode that consumes less than 1 milliwatt of power. Devices that cannot achieve this power level with an internal clock-gated state should implement power-gating through D3cold.

Network and radio devices are the notable exception to the 1-milliwatt guideline. Network and radio devices might require more power to maintain a connection to the network or to listen to wireless devices. Some system designers refer to these power-state transitions as *runtime D3* (RTD3).

Directed power management for PCIe devices

PCIe cards outside the SoC must enable a directed power management mechanism called Device-S4 in order to ensure that they can enter a low power mode. Without Device-S4, if a user plugs a device into a PCIe Root Port with user-accessible slots on a desktop Modern Standby system, and the driver for the device does not support runtime D3 (RTD3), the PCIe device may prevent the system from entering DRIPS. To avoid this problem, OEMs must opt in the Root Ports of the PCIe device to Device-S4. For Device-S4 to be engaged for a given PCIe device, the following requirements must be met:

1. The parent PCIe Root Port for the device must be specified as a constraint for DRIPS.
2. The parent PCIe Root Port must apply fundamental device reset for the all children downstream of the Root Port upon the Root Port's D0->D3 transition and deassert fundamental device reset to those children upon the D3->D0 transition. For more information about PCIe fundamental resets, please refer to Section 6.6.1 of the PCI Express Base Specification. The application of fundamental reset can be provided by supplemental ACPI mechanisms. For more information, please refer to this guide on PCI Power Management. To indicate that the platform honors this fundamental reset requirement, the firmware must define _DSD with support for GUID {FDF06FAD-F744-4451-BB64-ECD792215B10}. Without this, Directed DRIPS will not be triggered for devices under that PCIe Root Port. For more information, refer to the ACPI Device Specific Data (_DSD) for PCIe Root Ports.
3. Power resources must be unique to a given PCIe Root Port, i.e. not shared with any other Root Port.

Please note that PCIe cards that require Compatibility Support Modules (CSM) will not work in Modern Standby systems. CSM is not supported in the BIOS on Modern Standby systems due to the UEFI Secure Boot requirement. For more information, please refer to the Windows Hardware Compatibility Program Specification.

For more information about power management for particular device classes, system designers are encouraged to review Device-specific power management for modern standby, as well as device-specific documentation on Microsoft Collaborate.

Powering down networking devices

Powering down the networking and radio devices is another key part of preparing the hardware for low-power operation during sleep. Networking and radio devices are different from other devices outside of the SoC because they must remain on with power applied to listen for interesting events and wake the SoC. For example, the Wi-Fi radio must be capable of listening for packets that match WoL patterns and waking the SoC when a matching packet is detected.

Windows transitions networking devices to the D2/D3 state during sleep if they are expected to wake the SoC. The Windows network stack will configure WoL patterns and protocol offloads before placing the device in the low-power D2/D3 state. All networking devices, including Wi-Fi, mobile broadband (MBB), and wired Ethernet, must be capable of entering the D2/D3 state during sleep. If a networking device is not required to wake the system, Windows will transition the device to the D3 state. The networking device can be placed in the D3 state if the user has enabled airplane mode or disabled the specific networking device.

Each device has a different physical method to wake the SoC from its lowest power mode. Networking devices on SDIO or UART are expected to signal a GPIO line to wake the SoC. Networking devices connected via USB or HSIC are expected to use in-band USB resume signaling to wake the SoC. Networking devices on PCI or PCIe buses are expected to use in-band PME signaling to wake the SoC.

In addition, a radio device, such as a Bluetooth or near-field communications (NFC) device, is expected to transition to the D2 state if the user has turned on the radio for this device. In the D2 state, a Bluetooth radio listens for input events from paired mice and keyboards. If an input event is detected, the Bluetooth radio toggles a GPIO line connected to the SoC, which causes the SoC to wake up from its low-power mode.

Each networking or radio device has its own implementation of powering down for modern standby. System designers are encouraged to read the device class-specific documents on the Microsoft Collaborate website.

Powering down on-SoC host controllers

After all devices outside of the SoC, including network and radio devices, have been powered down, the host controllers to which the devices are attached must power down. Common host controllers include USB, PCI, SDIO, GPIO, and I^2C.

The driver for each host controller can power down the hardware only after every device connected to the host controller has powered down. A common example is the USB host controller. The USB host controller can power down only after all of the USB devices connected to it have entered a selective suspend state. If a USB host controller has a USB mouse and keyboard connected, the host controller can power down only after both the mouse and keyboard have powered down. If either the mouse or the keyboard stays powered on, the USB host controller also stays powered on.

All host controllers on the SoC must power down for sleep in order for the SoC itself to power down. This is why it is critical that every device perform device power management. The SoC itself can only power down when each host controller has powered down. The host controllers can power down only after all devices connected to them have powered down.

Powering down CPUs and GPUs

In terms of power management, CPUs and GPUs on the SoC chip are different from other devices. CPUs and GPUs power down as part of powering down the SoC itself and are able to be powered down whenever there is no software activity targeted to them.

Most software activity on the system will be stopped through the preparation stages detailed in Prepare software for modern standby. Microsoft Store apps will be paused as part of the PLM phase. Desktop applications will be paused as part of completing the DAM phase. The only CPU activities that remain after the platform enters the resiliency phase are idle operations of Windows itself. Similarly, there is little GPU activity because all apps have been paused and the screen is turned off.

Windows continually manages the power state of the CPUs on the system, even when the screen is on and the user is working with the PC. The same CPU power state management places the CPUs in a low-power mode during sleep. When all CPUs are in a low-power mode and all host controllers on the SoC have been powered down, the SoC itself can be powered down.

Powering down the SoC

When all the individual host controllers, CPUs, and GPUs on the SoC have been powered down, Windows will determine if it is safe to power down the entire SoC itself. The SoC vendor provides a power engine plug-in (PEP) to tell Windows when all of the state on the SoC has been saved so that the SoC is ready to enter a low-power mode. For Intel based SoCs, the PEP is provided inbox.

Each SoC vendor has a different implementation of a SoC-wide low-power state. These states are typically either clock-gated or power-gated states in which memory contents are preserved in self-refresh and the system can be awakened by a programmable timer and a small number of GPIO pins that consume very little power. Windows refers to the lowest SoC power state as deepest runtime idle platform state (DRIPS).

The DRIPS state always has the following characteristics:

- DRIPS is the lowest power consumption state for the SoC in which memory is preserved in a self-refresh mode.
- DRIPS allows the SoC to wake on events from networking, radio, and input devices.
- No CPU code is allowed to run during the DRIPS state.
- When the SoC is in the DRIPS state, the platform is consuming the least amount of power possible during sleep (with the exception of variances in power consumption caused by networking and radio devices).

Directed power management

Article • 03/19/2021

Directed Deepest Runtime Idle Platform State (DDRIPS) refers to mechanisms by which the OS directs devices to enter their appropriate low-power idle states when the system goes idle, thereby enabling the system to enter low power more reliably.

DDRIPS applies only to Modern Standby systems and targets only those devices that are constraints for DRIPS and their child devices, if any. Child devices can be devices that are directly enumerated by the parent device or that have a power relation with the parent device.

DDRIPS comprises two mechanisms, Directed PoFx (DFx) and Device-S4. For device stacks that support both mechanisms, both mechanisms can be used to power down different devices in the device tree depending on which mechanisms each individual device supports.

DFx

DFx is an extension to PoFx that enables the power manager to direct a device to enter a lower power state. DFx takes effect after a DRIPS constraint device has been active for at least two consecutive minutes (driver-configurable timeout, default 2 minutes) when the system is in Modern Standby and there is no activator-brokered software activity running. DFx directs devices to enter their target D-state for Modern Standby with the ability to arm for wake as they may during a normal Modern Standby transition. It does not carry the assumptions associated with an S-IRP, e.g. device reset is not required. DFx will not power down paging or debug devices.

DFx unwinds for Wait/Wake IRPs and PnP operations.

DFx is enabled on both AC and DC power.

Device-S4

Device-S4 provides a means to leverage existing S-IRP logic in drivers to initiate a D-state transition. Device-S4 takes effect after a DRIPS constraint device has been active for at least five consecutive minutes when the system is in Modern Standby and there is no activator-brokered software activity running. Device stacks must advertise a means to be reset or power cycled to be included in the Device-S4 transition.

Just like a normal S4 transition, Device-S4 a global operation where all applications and services are suspended and system functionality is akin to S3/S4. However, the S4-IRP is sent only to the offending device or device subtree in the device hierarchy. From the perspective of the device receiving a Device-S4, it is indistinguishable from a normal S4 transition. The device will enter the same D-state as during an S4 transition.

Device-S4 is enabled by default for USB devices, and it can be enabled via opt-in for PCIe plug-in devices on desktops. Keyboards and mice are excluded from Device-S4 in order to preserve their ability to wake from Modern Standby.

On desktop form factor systems, Device-S4 is enabled on AC power. On mobile form factor systems, Device-S4 is enabled only on DC power.

See also

- Introduction to the Directed Power Management Framework
- Directed power management for PCIe devices

Activators

Article • 10/24/2022

Windows can leave the System on a Chip (SoC) in the active power state during sleep while app background tasks are executing or critical system tasks are executing. Windows components that are allowed to leave the SoC in the active power state are called activators because they are registered with the power manager as capable of blocking the transition back to the idle power mode.

The most common activator is called Background Infrastructure (BI). BI is the system component that is responsible for managing and executing background tasks for Microsoft Store apps. BI enables its activator reference whenever a background task is executing during standby. As soon as all background tasks have stopped executing, or they are forced to stop executing, BI will release or clear its activator reference and the SoC can go back to the idle power mode.

Whenever an activator has an outstanding reference, the SoC will remain in the active power mode and will not enter the idle (DRIPS) power mode. There are a number of activators in Windows. The following activators are the most common.

Expand table

Activator name	Activator description	Reference enabled when...	Reference cleared when...	Notes
BI	Background Infrastructure. This activator manages background tasks for Microsoft Store apps.	Any background task for a Microsoft Store app is running.	When all background tasks for Microsoft Store apps have completed.	
WU	Windows Update. This activator is part of the built-in Windows Update service.	Windows Update is scanning for critical downloads or installing critical updates.	Windows Update is not performing any activity.	
WNS	Windows Notification Service. This activator	Windows is connecting to WNS. When the system needs to	When there is no WNS activity or active push notifications.	This component is shared with the Image Download

Activator name	Activator description	Reference enabled when...	Reference cleared when...	Notes
	manages the system connection to the WNS service in the cloud.	deliver a push notification to a Windows component or Microsoft Store app.		Manager. WNS supports delivery of notifications to many built-in Windows system services.
PLM	Process Lifetime Manager. This activator manages the execution and scheduling of Microsoft Store apps.	The PLM is waiting for a Microsoft Store app to suspend.	All Microsoft Store apps are suspended.	The longest typical duration is five seconds. The PLM reference will be enabled when a Microsoft Store app is playing music with the screen off.
NCSI	The Network Connectivity Status Indicator activator manages system-wide Internet availability.	Underlying network connectivity has changed. For example, the reference will be briefly enabled after Wi-Fi loses connectivity to an access point.	No network state changes are occurring.	
Group policy	This activator manages application of group policy settings configured by the IT administrator.	When group policy settings are being applied.	When there are no group policy settings to apply.	This activator will only be enabled if the system administrator or IT administrator has applied new group policy settings to this system.
Audio Service	Takes a timer request in PDC to make sure when there is any active audio stream that the system maintains	Any App playing Audio.	App stops playing Audio.	

Activator name	Activator description	Reference enabled when...	Reference cleared when...	Notes
	regular timer resolution.			
TCPIP	Activator for the TCP/IP stack in Windows.	The TCP/IP stack is performing Router Solicitation or Duplicate Address Detection.	The TCP/IP stack is finished performing Router Solicitation or Duplicate Address Detection.	
DHCP	Dynamic Host Configuration Protocol Client. This activator handles dynamic IPv4 address acquisition on the local network.	The device has connected to a new network and needs to acquire a new lease for an IP address, or when it needs to renew a lease on an existing IP address.	The device has finished its address lease operation.	
Universal Telemetry Client	Connected User Experiences and Telemetry service. This activator is used to send diagnostic data and feedback to Microsoft (used to improve Windows and keep it secure, up to date, and working as expected) and to download diagnostic data collection configuration settings.	The Connected User Experiences and Telemetry service begins work to download configuration settings or send diagnostic data and feedback to Microsoft.	The Connected User Experiences and Telemetry service completes work to download configuration settings or send diagnostic data and feedback to Microsoft.	Diagnostics, feedback, and privacy in Windows
System Maintenance Service	This activator supports application	A Microsoft Store app is being prelaunched.	When no Microsoft Store apps have	Apps are prelaunched only when the

Activator name	Activator description	Reference enabled when...	Reference cleared when...	Notes
	prelaunch scenarios.		requested to be prelaunched and no apps are currently being prelaunched.	app itself has requested it and the system detects it is a frequently used app that would lead to user perceivable launch performance improvements.
Host Activity Manager	This activator manages a variety of scenarios where Microsoft Store apps perform user perceivable function.	A Microsoft Store app is playing audio in the background.	When no Microsoft Store apps are playing audio in the background.	This does not encompass all scenarios where this activator reference is enabled.
BITS Service	Activator for the Background Intelligent Transfer Service, a Windows service for transferring files to and from HTTP and SMB servers in the background. Background Intelligent Transfer Service	BITS begins a file transfer job.	BITS completes a file transfer job.	
Update Orchestrator Client	This activator is part of the built-in Windows Update Orchestrator service.	Orchestrator is scanning, downloading, installing, or committing updates.	Orchestrator is not performing any activity.	
Windows Error Reporting	Windows Error Reporting. This component is responsible for	WER begins to upload a failure report.	Report upload has either finished or reached the	If the upload has started, WER allows for maximum 1 hour

Activator name	Activator description	Reference enabled when...	Reference cleared when...	Notes
	uploading crash, hang and other failure and diagnostic reports to an analysis service.		maximum allowed time.	(user mode reports) or 2 hours (kernel mode reports) after the connected standby is initiated.
Location Client	Used by the Windows location service during positioning scans and location upload.	Position is requested using Bluetooth, Cellular, or Wi-Fi scans as well as when location upload is occurs.	Positioning scan or location upload has completed.	
Find My Device	Find My Device. This activator periodically tracks the device's locations on a map to help find the device if users think they have lost it.	When Location and Find my device option in windows settings are enabled.	When Location or Find my device option in windows settings is not enabled.	Users have to sign in with their Microsoft account as an administrator to use or change this setting. If the user have connected to a work or school account, Find my device won't work.
Voice Activation	This activator is for voice activation (keyword spotting) of assistant applications.	The system is requested to process a keyword spotting detection or actively monitor keyword spotting.	No keyword spotting is being processed and no keyword spotting needs to be actively monitored by the system.	Speech, voice activation, inking, typing, and privacy
WP OmaDmClient Client	OMADMClient is an exe which is core server pipe and it provides device management to an mobile device	OMADMClient begins talking to the MDM server	OMADMClient ends talking to the MDM server	

Activator name	Activator description	Reference enabled when...	Reference cleared when...	Notes
	management (MDM) server. The work happens on client devices when the user is enrolled into Settings->Accounts->Access Work or School or signs in with corporate credentials in OOBE			

Activators are system components and their runtime and behaviors cannot be changed by system hardware designers.

Activator references and their durations are displayed in the built-in SleepStudy diagnostic utility. Individual references may be observed through instrumentation and tracing using the Windows Performance Analyzer (WPA).

Modern standby user experiences

Article • 01/29/2021

Modern standby is more than just a low-power sleep state—it is part of many Windows 10 fundamental user experiences. When the screen is off, the modern standby state is presented to the user as the system "sleep" state.

The following topics describe the modern standby user experience.

Topics	Description
Transitioning between idle and active states	A key feature of Modern Standby is the ability of the SoC to quickly transition between idle and active states. This allows a device to maintain a low power floor while also servicing value-adding activity and waking quickly.
Turning off	The fundamental user experience of modern standby is simply putting the PC to sleep.
Wake sources	One of the primary benefits of Modern Standby is the instant resume when the system is turned on. Modern Standby systems resume instantly in response to several wake sources.
Simple app and tile updates	Modern standby enables the system to stay fresh and up-to-date by being able to stay connected to the network during sleep and by carefully controlling the execution of app background tasks.
Audio notifications	If a lock screen app generates a notification and resulting toast during modern standby, a short audio alert may be played depending on system form factor and lid state.
Focus assist	Notifications during modern standby and their auditory alerts can be disturbing to some users, particularly in the evening and late night hours. The Focus assist feature suppresses notifications and helps save battery power. Focus assist allows the user to define a period of time each day during which notifications and background tasks are suppressed.
Network connectivity	With Adaptive Connected Standby (ACS), Windows provides connectivity during sleep on AC power. On DC power, Windows provides network-connected experiences during Modern Standby when the user opts in, while otherwise quiescing networking activity to preserve battery life. Regardless of power source, reconnection is instant upon resume from standby. Under some conditions, mobile hotspot connections also remain available during Modern Standby.

Transitioning between idle and active states

Article • 06/24/2021

During sleep, Windows periodically transitions the SoC platform from idle mode to active mode to perform value adding activities such as receiving an email notification or performing a maintenance task. Additionally, on-demand transitions to active mode can occur in response to user inputs, interrupts from networking devices, and other hardware events.

After the SoC has entered the DRIPS state, the platform will typically be in its lowest power mode. During sleep, the platform will switch between this idle (DRIPS) mode and the active mode periodically. Low power is achieved in sleep by spending the majority of the time in the DRIPS mode and very small amounts of time in the active mode.

Transitioning from idle to active

Windows will transition the SoC to the active mode for the following reasons:

- An on-demand wake event due to an interrupt from a networking device (Wi-Fi, mobile broadband, or Ethernet). The networking device generates an interrupt when a matching WoL pattern is detected or when the media or connection state changes. The interrupt wakes the SoC using an out-of-band GPIO, in-band USB resume signaling, or PCI PME interrupt.
- An on-demand wake event due to a platform device interrupt from a source such as the battery fuel gauge, battery charge controller, or thermal sensors.
- An on-demand wake event due to an interrupt caused by user input at a keyboard, touchpad, external USB peripheral, power button, or lid switch.

As soon as the SoC wakes and the platform exits the DRIPS state, the CPUs start running code again. However, the screen stays powered off unless the interrupt was a result of user input or connecting to a power source. The driver for the device that causes the interrupt is responsible for reporting user presence to the OS if the interrupt corresponds to user input. The Windows power manager determines whether the screen should be powered on; the system designer needs only to ensure that the interrupt from each device is connected and programmed to wake the SoC from the DRIPS state.

System designers must work closely with the SoC vendor to connect device interrupts for SoC wakeup. Some SoC devices require device interrupts to be connected to special "wake-capable" inputs or configured for wake using system firmware.

Transitioning from active to idle

Windows transitions the SoC platform from active mode to idle mode after all software activity is stopped and the devices on and off the SoC chip have entered low-power states.

The SoC transitions between idle and active modes during sleep based on the triggers explained in the previous section. Whenever the SoC is active, Windows will aggressively attempt to return it to the idle (DRIPS) state to reduce power consumption.

The SoC will also be active when the system exits modern standby in response to user input, such as a power button press. The SoC will be put back in the idle (DRIPS) state automatically when all of the following conditions are true:

- All devices outside of the SoC have been powered down.
- All network and radio devices have entered their low-power state to wait for packets matching WoL patterns or wake interrupts.
- All host controllers on the SoC have been powered down.
- All app background tasks have completed.
- All CPU and GPU activity has stopped and all CPUs are idle.

Almost all modern standby power problems are related to making sure all of these five conditions have been met. Windows includes a built-in diagnostic tool called SleepStudy to help discover which of these conditions is not being met when the SoC is prevented from entering the idle (DRIPS) state.

Turning off

Article • 03/11/2020

One fundamental user experience of Modern Standby is simply putting the PC to sleep. The system will enter Modern Standby when the user does any of the following:

- Presses the system power button.
- Closes the lid or tablet cover, or closes the tablet into an attached dock.
- Selects Sleep from the power button on the Windows Start menu.
- Waits for the system to idle and enter sleep automatically, according to the Power and sleep settings.

The user experience for turning off the Modern Standby PC is instant. The display is immediately powered off when the user presses the power button or does something in the preceding list. The system might still be active for a few seconds after the power button is pressed and the screen turns off, but the experience is smartphone-like and instant. While the system appears to the user to be asleep, it can transition between different hardware and software power modes.

Related Articles

- Transitioning between idle and active states
- Prepare hardware for Modern Standby
- Prepare software for Modern Standby
- Activators

Modern Standby Wake Sources

Article • 01/07/2025

A PC supporting the Modern Standby power model must wake from standby in response to specific events, even in a low-power idle state.

This topic details the wake sources that must wake the processor and explains which events should turn the screen on or keep it off. System integrators should use this information to ensure their hardware, firmware, and software can configure wake sources appropriately.

Overview of Modern Standby User Experience

The Modern Standby experience is designed to mimic a cellular phone. When users finish using their phones, they press the power button, and the phone enters sleep mode. It remains asleep until the user presses the power button again or receives a call, email, or message.

Similarly, a PC in Modern Standby appears off—the screen is blank, there are no visible LEDs, and no acoustic noise. However, it remains on and connected to the Internet, using any available network connection (Wi-Fi, mobile broadband, or Ethernet). The Modern Standby PC also has long battery life in its screen-off state, similar to a cell phone.

Achieving the Modern Standby experience requires all devices and software in the PC to participate in system power management. Long standby battery life is achieved by allowing all devices and the core silicon or System on a Chip (SoC) to enter a low-power idle state. The networking subsystem stays connected to wake the system for incoming emails or VoIP calls. The real-time nature of Modern Standby is enabled by platform devices waking the SoC for the correct events at the correct times.

All wake sources in a Modern Standby PC must wake the SoC from its deepest idle state. Some wake sources should generate a wake signal that turns on the display. The difference between waking the SoC and turning on the display is crucial for the Modern Standby experience. The following rules govern platform wake behavior:

- Wake source operation and scenarios are consistent across all Modern Standby PCs, regardless of processor architecture (x86, x64 or Arm).
- Wake source operation may differ based on whether the system is on AC or DC power. Differences are noted in the tables below.

- Some wake sources are hardware-dependent, e.g., wake on fingerprint or optical disc drive ejection. Modern Standby systems should support all wake scenarios for which they have the necessary hardware.

> **Note**
>
> Starting in Windows 11, version 24H2, a new power-saving measure was introduced to Modern Standby to prevent unexpected battery drain. If excessive battery drain is detected, most wake sources will be disabled. In this state, the device can be woken from Modern Standby by pressing the power button or opening the lid.

The remainder of this topic describes the different types of wake sources, including whether they can turn on a device's display, whether they are enabled by default, and any differences in operation based on AC or DC power.

Most systems enable **Adaptive Connected Standby** by default.

Types of Wake Sources

Always-on timer

The core silicon or SoC chip in a Modern Standby platform has one or more timers that are always powered on that so that Windows can schedule future work and place the SoC into a deep idle state. During standby, the always-on timer reliably wakes the SoC as programmed by the OS.

Buttons and Lid

The system power button is a common user-initiated wake source in a Modern Standby platform. All Modern Standby PCs must ensure the power button can always send a wake interrupt to the SoC. To deliver an instant-on experience, the power button must wake the SoC from the deepest idle state without delay. Another common wake source on clamshell devices is opening the lid, which wakes the SoC. The Windows button can also wake the SoC.

Communications Devices

Wi-Fi, Ethernet, and mobile broadband (MBB) devices provide continuous connectivity in Modern Standby. They enable wake sources like Universal Windows Platform (UWP) application notifications and syncing, as well as Bluetooth notifications and syncing.

When Disconnected Standby, certain **applications** in above will not have connectivity in standby states. However, this does not apply to specific system processes, which can remain active (e.g., DHCP) and use the network during Disconnected Standby.

When using a DC power source, the networking stacks may initiate disconnection from networks. This includes L2 disconnect for Ethernet, disconnection of Wi-Fi and Bluetooth, and turning off the MBB radio to save power consumption.

Input Devices

Using **HIDI2C** for input peripherals is recommended for better power efficiency, but not required. USB can be used to connect input devices like TouchPads, touch digitizers, or pen digitizers. A precision TouchPad should be capable of waking the system from deep idle, regardless of connection type (USB, I2C, or SPI). Pen digitizers must not wake the system from deep idle.

In addition to chassis buttons, a Modern Standby PC might have other input devices integrated into or attached to the system. User input through these devices must always wake the SoC from the deepest idle state and turn on the display, unless the lid is closed and input suppression is engaged.

Starting in Windows 11, touch screens may also wake the system from deep idle or S3, though it is not required. OEMs, ODMs, and system implementers should consider the additional battery drain of a wake-capable touch screen in standby mode.

Voice Input

On systems with a hardware keyword spotter (HW-KWS), voice input of a specific keyword can wake the SoC from the deepest idle state and turn on the display. Systems with Multiple Voice Assistants (MVA) support and an MVA-capable driver can wake the SoC and turn on the display with additional voice assistant commands.

Insertion or Removal of a Connector or Device

Inserting or removing certain connectors or devices can wake the SoC from the deepest idle state and turn on the display.

Windows Update

Windows Update can always wake the SoC from the deepest idle state to scan for updates. It can also wake the SoC to download and install updates and restart a device,

depending on the power source.

Universal Windows Platform (UWP) Applications

UWP applications can wake the SoC from the deepest idle state to sync and display notifications, depending on the power source, notification priority levels, and app-specific user configurations.

Remote Access

Remote Desktop and File Sharing can wake the SoC from the deepest idle state, depending on the target device's power source and network connection. Remote Desktop can turn on the display on the target system.

Audio

Audio from internal or Bluetooth speakers can wake the SoC, allowing audio to continue playing even when the screen is off.

Environmental Context Changes

The Modern Standby PC must respond in real-time to changes in environmental conditions, such as thermal events and power source changes.

Wakes SoC and can turn on display

Buttons and Lid

Expand table

Device	Turns on Display (AC Power)	Turns on Display (DC Power)	Remarks
Power Button	Yes	Yes	The Windows power manager activates the display when the power button is pressed.
Windows Button	Yes (if wake-enabled)	Yes (if wake-enabled)	The Windows power manager activates the display when the Windows button is pressed. The button is considered user input.

Device	Turns on Display (AC Power)	Turns on Display (DC Power)	Remarks
Lid Switch (mechanical or sensor-based)	Yes	Yes	Various types of lid switches (mechanical or sensor-based) are treated similarly by Windows. These switches can turn off the display when a tablet is docked or covered. Opening the lid or cover, or adjusting the display, will turn on the display. The Windows power manager handles this automatically.

> **Note**
>
> Starting from Windows 11, version 24H2, pressing the power button will engage input suppression. The display will not turn on in clamshell mode if the lid is closed, unless an external display is connected.

Input Devices

Expand table

Device	Turns on Display (AC Power)	Turns on Display (DC Power)	Remarks
Integrated Keyboard (HIDI2C)	Yes (See Note)	Yes (See Note)	The display turns on with keyboard input. All keys, except volume buttons, must generate a wake interrupt. Consumer control keys must also generate a wake interrupt. Note: In convertible tablets, the keyboard should not wake the SoC if not visible.
External USB Keyboard	Yes (See Note)	Yes (See Note)	Multiple key presses might be needed to generate a resume event. Note: Keyboard wake may be disabled to save battery.
External Bluetooth Keyboard	Yes (See Note)	Yes (See Note)	Note: Keyboard wake may be disabled to save battery.
Integrated Precision	Yes (See Note)	Yes (See Note)	Finger movement or button activation on the TouchPad should cause a wake event. Note: TouchPad wake is optional. In convertible tablets,

Device	Turns on Display (AC Power)	Turns on Display (DC Power)	Remarks
TouchPad (I2C, USB, or SPI)			the TouchPad should not wake the SoC if not visible.
External USB Mouse	Yes (See Note)	Yes (See Note)	Pressing any button should generate a resume event. Multiple button presses might be needed. Note: Mouse wake may be disabled to save battery.
External Bluetooth Mouse	Yes (See Note)	Yes (See Note)	Pressing any button will generate a resume event. Movement wake is optional. Note: Mouse wake may be disabled to save battery.
Fingerprint Reader	Yes	Yes	Note: Fingerprint reader wake may be disabled on DC power to save battery.
Touch Screen	Yes (See Note)	Yes (See Note)	Note: Starting in Windows 11, touch screens can wake the system from Sleep. This is optional and may increase battery drain.

> **Note**
>
> The display will not turn on in clamshell mode if the lid is closed, as input suppression is activated when on DC power or AC power starting from Windows 11, version 24H2. However, if an external display is connected while the lid is closed, the system will not engage input suppression. For more information, see input suppression.

Voice Input

Expand table

Event	Turns on Display (AC Power)	Turns on Display (DC Power)	Remarks
Voice Input (Multiple Voice Assistants)	Yes, if Wake on Voice-capable and enabled by user.	Yes, if Wake on Voice-capable, with hardware keyword spotter and MVA-capable driver. If only software keyword spotter, Wake on Voice is off by default on DC power.	Wake on Voice may be disabled on DC power to save battery.

> **Note**
>
> Beginning with Windows 11, version 24H2, Voice Input will no longer be supported to wake the device from Sleep.

Connector or Device Insertion/Removal

Expand table

Device	Turns on Display (AC Power)	Turns on Display (DC Power)	Remarks
Dock Attachment/Removal	Varies	Varies	Depends on the devices in the dock and their state. Dock attachment should be treated as individual device attachment. Detection of new devices in the dock should wake the SoC.

Windows Update

Expand table

Event	Turns on the display - AC power	Turns on the display - DC power	Remarks
Restart	Yes	No. Restart for Windows Update is disabled on DC power.	

Universal Windows Platform (UWP) Applications

Expand table

Event	Turns on the display - AC power	Turns on the display - DC power	Remarks
Skype: incoming calls and IMs	No longer supported starting with version v8.61	No longer supported starting with version v8.61	
3rd party VoIP calls and IMs	Depends on the type of notification.	Depends on whether the app is exempt, the notification priority level, and the type of notification.	See Note.

Event	Turns on the display - AC power	Turns on the display - DC power	Remarks
Bluetooth device notification	Depends on the type of notification.	No	See Note.
Bluetooth authentication: proximity	Depends on the type of notification.	No	See Note.
Location services (Geofencing APIs trigger push notification)	No	No	See Note.
Location services (Geofencing APIs trigger push notification)	No	No	See Note.
Other UWP apps: push notifications	Depends on the type of notification.	Depends on whether the app is exempt, the notification priority level, and the type of notification.	See Note.

> **Note**
>
> If the device is connected to AC power, the notifications of all priorities (high, medium, or low priority) deliver for the UWP apps. If the device is on DC (battery) power, the notifications of all priorities deliver only for the exempt UWP apps. Different types of notifications have different levels of priority, depending on the specific app. For example, the Inbox Mail app considers a notification of incoming mail as low priority. If the notifications are being delivered during Modern Standby, only toast notifications will turn on the display. For more information about UWP app notifications, see WNS Notification Priorities.

To set an app as exempt, right-click **Start**, and then select **Settings**>**System**>**Battery**. You can see the apps that consume high battery usage. If you want to exempt an app, select the app. If **Let Windows decide** is turned on, select **Always** to allow the app to run in the background. To allow notifications from the app, select **Settings**>**System**>**Notifications & actions**, and then turn on the app in **Get notifications from these senders**.

To configure location services, go to **Settings**>**Privacy**>**Location**. Location services, including apps using Geovisits and Geofencing APIs, are not functional during Modern Standby.

Remote Access

Expand table

Event	Turns on the display - AC power	Turns on the display - DC power	Remarks
Remote Desktop	Yes	Yes, if the target system has an Ethernet connection.	

Environmental context changes

Expand table

Event	Turns on the display - AC power	Turns on the display - DC power	Remarks
Connecting a power source	Yes (on AC attach)	No (on AC detach)	The Windows power manager will turn on the display when the battery subsystem has indicated AC power has been connected. The GPIO interrupt for power source changes must cause the ACPI _PSR method under the power supply device to be executed. The power subsystem must wake the SoC any time the power source changes, including when the system is attached or removed from a dock that has a battery or AC power source. After AC power is connected, the display will remain on for five seconds, unless there is input to the system during this five-second window.

Wakes SoC but cannot turn on display

Expand table

Device	Remarks
Always-on timer	Each SoC has a different mechanism for programming the always-on timer.

Expand table

Device	Remarks
SD card insertion/removal (SDIO controller-attached)	
SD card insertion/removal (USB-attached)	The SD controller selected must be capable of detecting card insertion and removal while in the USB suspend state drawing less than 1 milliwatt average.
Headphone or microphone insertion/removal	Attaching a headphone or microphone to the system provides an interrupt to enable the audio stack to correctly route audio.
eSATA insertion/removal	

Expand table

Device	Remarks
Wi-Fi radio	The Wi-Fi wake source is not required for Disconnected Standby systems.
Mobile broadband (MBB) radio	
Bluetooth radio	Windows and its drivers are responsible for detecting the type of associated Bluetooth device. If a keyboard, mouse, or other user-input device is responsible for causing the Bluetooth radio to wake the SoC, the display will turn on. Other Bluetooth devices such as portable audio headphones will not cause the display to turn on.
Wired LAN (USB-attached, Modern Standby-capable)	Wired LAN devices in Modern Standby platforms or their supported docks must support pattern-match offloads in order to be Modern Standby-capable. Wired LAN devices connected over USB may undergo a Device-S4 transition during Modern Standby and lose wake capability.

> **Note**
>
> Windows can turn on the display when an incoming critical alert or activity is detected over the network. Examples include high priority notifications from lock-screen applications and VoIP calls.

Windows Update

Expand table

Event	Enabled by default - AC power	Enabled by default - DC power	Remarks
Scan	Yes	Yes	
Download	Yes	Yes, but only limited to interactive download. All non-interactive downloads are paused. No new downloads will be initiated.	
Install	Yes	No. In-progress installs get paused. Does not kick off new install.	

Remote Access

Expand table

Event	Enabled by default - AC power	Enabled by default - DC power	Remarks
File Sharing	Yes	No	Please refer to File sharing over a network in Windows 10 for more information.

Universal Windows Platform (UWP) Applications

Expand table

Event	Enabled by default - AC power	Enabled by default - DC power	Remarks
Background download of content	Yes	No, unless the app is exempt.	See Note.
Inbox Mail app: mail sync notifications	Yes	No, unless the app is exempt.	See Note.
Inbox People app: contact sync notifications	Yes	No, unless the app is exempt.	See Note.
Inbox Calendar app: calendar sync notifications	Yes	No, unless the app is exempt.	See Note.
Sync with Bluetooth devices	Yes	No	

Event	Enabled by default - AC power	Enabled by default - DC power	Remarks
App operations that require network (via SocketActivityTrigger)	Yes	Yes	
OneNote and VoiceRecorder: background audio recording	Yes	Yes	

> **Note**
>
> If the device is connected to AC power, the notifications of all priorities (high, medium, or low priority) deliver for the UWP apps. If the device is on DC (battery) power, the notifications of all priorities deliver only for the exempt UWP apps. Different types of notifications have different levels of priority, depending on the specific app. For example, the Inbox Mail app considers a notification of incoming mail as low priority. If the notifications are being delivered during Modern Standby, only toast notifications will turn on the display. For more information about UWP app notifications, see WNS Notification Priorities.

> To set an app as exempt, right-click **Start**, and then select **Settings**>**System**>**Battery**. You can see the apps that consume high battery usage. If you want to exempt an app, select the app. If **Let Windows decide** is turned on, select **Always** to allow the app to run in the background. To allow notifications from the app, select **Settings**>**System**>**Notifications & actions**, and then turn on the app in **Get notifications from these senders**.

Audio

Expand table

Event	Standby Entry Method	Enabled by default - AC power	Enabled by default - DC power	Remarks
Audio local and streaming playback (internal speakers) during screen off	Idle to Standby	Yes	Yes	

Event	Standby Entry Method	Enabled by default - AC power	Enabled by default - DC power	Remarks
Audio local and streaming playback (internal speakers) during screen off	Explicit entry to Standby (Power button press, closing the lid or through Start menu)	No	No	
Audio local and streaming playback (Bluetooth speakers) during screen off	Idle to Standby	Yes	Yes	Low Power Audio is not currently supported for Bluetooth speakers, so audio will play during Modern Standby but not be low power.
Audio local and streaming playback (Bluetooth speakers) during screen off	Explicit entry to Standby (Power button press, closing the lid or through Start menu)	No	No	Low Power Audio is not currently supported for Bluetooth speakers, so audio will play during Modern Standby but not be low power.

Environmental Context Changes

Expand table

Event	Remarks
Thermal event	All temperature sensors must wake the SoC from the deepest power state to indicate temperature change. ACPI firmware should monitor thermal zone temperature changes continuously during standby and when the SoC is in the deepest idle state. The ACPI firmware should report to the Windows thermal manager when the temperature rises above the trip points.
Battery charge completion	
Battery threshold change	The battery subsystem must wake the SoC from its deepest idle state anytime the remaining capacity goes below the value specified by Windows in the _BTP control method. The battery subsystem must wake the SoC from its deepest idle state anytime the remaining capacity goes below the value specified by DesignCapacityOfLow in the_BIX control method. Windows will hibernate (x86) or shut down (Arm) the system when the remaining capacity falls below DesignCapacityOfLow.

Event	Remarks
Unplugging from a power source	The GPIO interrupt for power source changes must cause the ACPI _PSR method under the power supply device to be executed. The power subsystem must wake the SoC any time the power source changes, including when the system is attached or removed from a dock that has a battery or AC power source.

Simple app and tile updates

Article • 05/25/2022

Modern standby enables the system to stay fresh and up-to-date by being able to stay connected to the network during sleep and by carefully controlling the execution of app background tasks.

An app can create a background task by using a specific set of WinRT APIs to enable the task to run on a regular schedule or in response to an incoming network event. WinRT is the Windows Runtime API surface and is distinct from Windows RT, which is the version of Windows that runs on the Arm processor architecture. The WinRT APIs for background tasks allow the app's runtime to be carefully managed by Windows to prevent excessive battery drain.

One common use of background tasks is to update the content of tiles on the Start screen. The content that an app displays in its tile can change in response to notifications. The tile can update in response to notifications in real time during modern standby as long as the the user has allowed the app to always run background tasks. The user can set this via Settings > Battery > See which apps are affecting your battery life, select the app, uncheck "Let Windows decide when this app can run in the background," and check “Allow the app to run background tasks.” On Arm systems, the tile will additionally update periodically during modern standby even if the app is not set to always run in the background. The user will see the updated tile as soon as the system turns on—that's because the app's background task ran during modern standby.

Apps can also use background tasks to check for new app-specific content during modern standby using a periodic timer. Timer expirations for an app that uses a periodic timer during modern standby are automatically coalesced with timer expirations for other apps. Timer expirations are limited to be no more frequent than once every 15 minutes, and when an app's background task runs, it has a very limited amount of CPU time available to do the update.

For more information about background tasks, see Launching, resuming, and background tasks.

Note During modern standby, desktop applications cannot run and cannot access the network. System services have limited runtime, but no network access.

Audio notifications

Article • 06/18/2024

If the device has idled to modern standby, and an app generates a critical notification (such as Alarm, Cellular, or VOIP), a short audio alert is played on desktop form factor systems and for mobile form factor systems if the lid is open. The display stays turned off while the alert plays unless it is associated with a toast notification that turns on the display.

If the user has muted audio through the Settings menu, the audio alert does not play. If modern standby was entered through an explicit user action (pressing the power button, closing the lid or through the start menu), the audio alert does not play.

Focus assist

Article • 02/07/2020

Notifications during modern standby and their auditory alerts can be disturbing to some users, particularly in the evening and late night hours. Focus assist allows the user to define a period of time each day during which notifications are suppressed and go directly to Action center. Users can configure this Focus assist setting via Settings > System > Focus assist > Automatic rules > During these times.

More information about configuring Focus assist settings can be found here.

Network connectivity

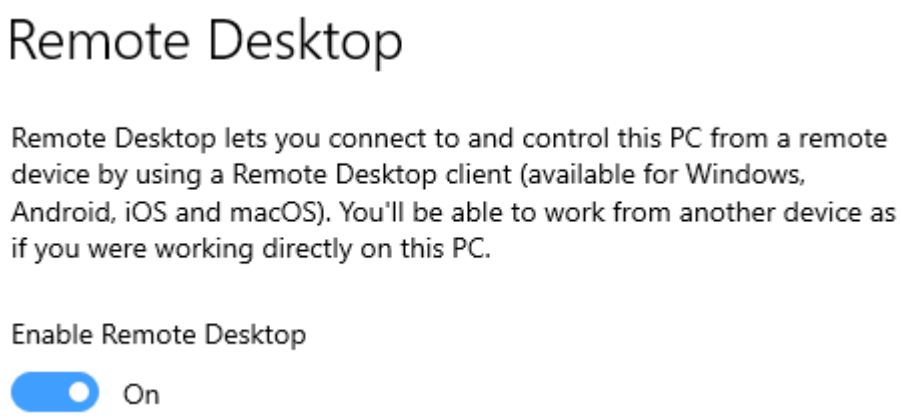
Article • 04/05/2021

Adaptive Connected Standby

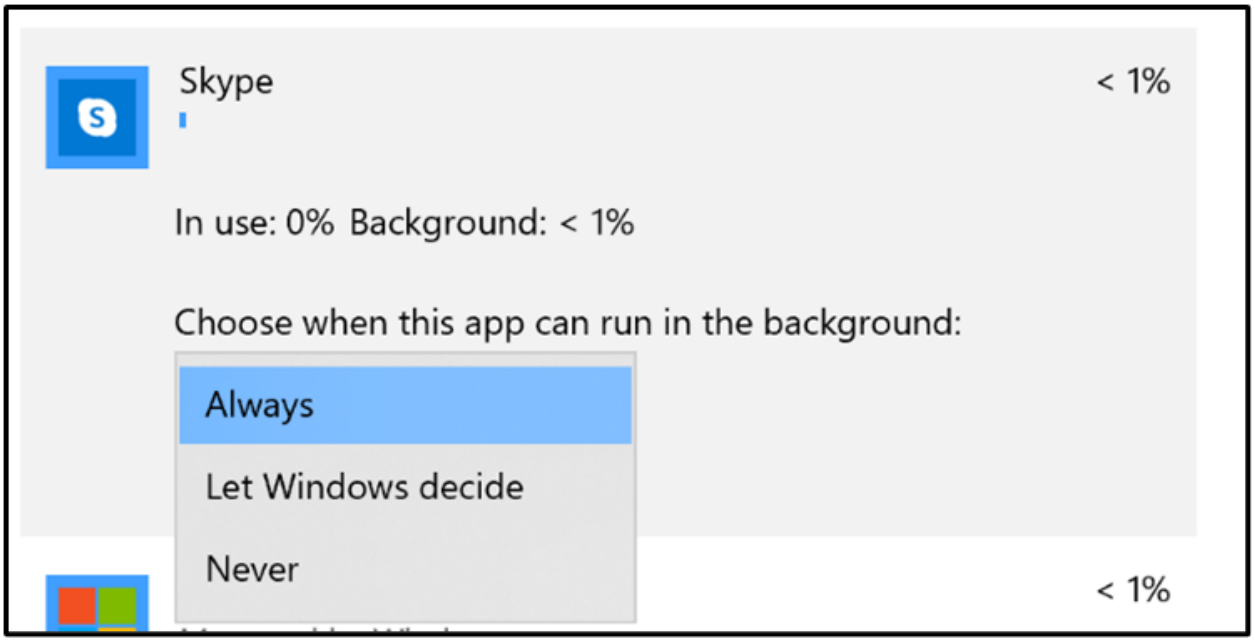
Starting in Windows 10 version 2004, Adaptive Connected Standby (ACS) is the connectivity default on Modern Standby systems. With ACS, Windows controls connectivity in sleep on Modern Standby systems while on battery power. Windows provides Internet connectivity to apps that require it and otherwise prevents networking activity during sleep. The network will be ready for immediate use when the system resumes from sleep. Internet connectivity refers to any combination of Ethernet, Wi-Fi and mobile broadband (MBB) that the system is capable of connecting to.

The user scenarios that require network connectivity in Modern Standby are Wake on Remote Desktop and receiving notifications from UWP apps. To maintain connectivity for these scenarios, the system will allow network activity during sleep if the user has enabled Remote Desktop or enabled any UWP app that has background tasks requiring network to always run in the background. Otherwise the system will quiesce network activity during sleep, while providing instant connectivity upon resume. This network connectivity optimization preserves the end user experience, while saving battery life by reducing networking activity when it is not needed.

Remote Desktop can be enabled via Settings > System > Remote Desktop > Enable Remote Desktop:

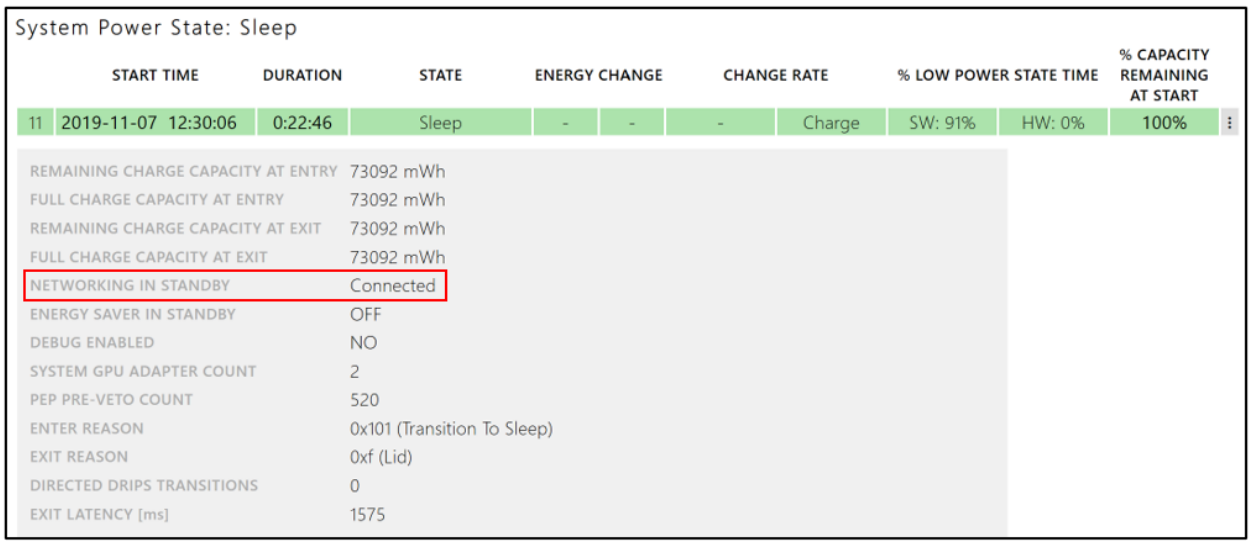

Remote Desktop
Remote Desktop lets you connect to and control this PC from a remote device by using a Remote Desktop client (available for Windows, Android, iOS and macOS). You'll be able to work from another device as if you were working directly on this PC.
Enable Remote Desktop
On

UWP apps can be enabled to always run in the background via Settings > System > Battery > Battery usage per app > Choose when this app can run in the background and selecting “Always” in the dropdown. By default, apps are set to “Let Windows decide,” which will not cause the app to remain connected.



ACS will determine the type of connectivity needed at the beginning of a sleep session and it stays the same for the entire session. For example, if the user enables an app that requires network access during sleep, the OS will determine that connectivity is needed at the beginning of a sleep session and stay connected while in sleep.

System Power Report (powercfg /spr) can be used to tell whether apps are connected or not in sleep during Modern Standby. This information is available in the “Networking in standby” field in session details:



Connectivity and system storage

Starting in Windows 10 version 2004, ACS is the connectivity default regardless of storage type (SSD, HDD or hybrid).

Windows Update and Wake on Voice

There is no change to the behavior of Windows Update and Wake on Voice scenarios during Modern Standby. Windows Update and Wake on Voice are able to access the network upon waking the system from sleep to perform updates and respond to voice commands, respectively.

Mobile Hotspot

The tables below outline the behavior of Mobile Hotspot on Modern Standby systems compared to Traditional Sleep (S3) systems.

Modern Standby

	User-initiated sleep entry (lid close, power button, Sleep from Start menu)		**Idle timeout**	
Power source	AC power	DC power	AC power	DC power
Hotspot status during sleep	Off (System achieves DRIPS)	Off (System achieves DRIPS)	On (Screen off)	On (Screen off, not limited by 5-minute timeout for power requests)
Hotspot status after resume from Modern Standby	Off	Off	On	On

Traditional Sleep (S3)

	User-initiated sleep entry (lid close, power button, Sleep from Start menu)		**Idle timeout**	
Power source	AC power	DC power	AC power	DC power
Hotspot status during sleep	On (Away Mode)	Off	On (Screen off)	On (Screen off)
Hotspot status after resume from S3	On	Off	On	On

Platform design for modern standby

Article • 05/19/2021

To modern connected standby, a PC hardware platform must meet a specific set of requirements. These requirements govern the selection of the SoC chip, DRAM, networking device, and other key hardware components.

Enabling modern standby on a PC platform requires careful planning and engineering. The primary reason for additional engineering is to deliver the low power consumption that the end user expects when the system is in a sleep state and the screen is turned off. Users will not tolerate excessive battery drain, particularly relative to the very good battery life of most smartphones.

The second largest engineering investment for modern standby is to enable low-power communications (Wi-Fi, mobile broadband, and Ethernet). Each communications device includes a significant amount of autonomous processing capability and firmware to allow the platform's SoC or core silicon to power off while maintaining connectivity.

Low-power core silicon (CPU, SoC, DRAM)

The modern standby power state requires frequent transitions between a low-power idle mode and short periods of activity. Through all these transitions, the system is in standby and the screen stays turned off. This model allows the operating system and apps to be always on and running while the hardware delivers low idle power. This combination results in low average power and long battery life during standby.

A modern standby platform with long battery life includes low-power core silicon (or SoC) and DRAM that have the following characteristics:

- The capability to switch between idle and active modes in less than 100 milliseconds. The active mode allows code to run on the CPU(s), but does not necessarily allow accessing the storage device or other host controllers or peripherals. The idle mode can be a clock-gated or power-gated state, but should be the state that has the lowest power consumption for the SoC and DRAM.
- DRAM technology and size to minimize power consumption while in self-refresh mode. Current modern connected standby PCs typically use mobile DRAM (LP-DDR) or low-voltage PC DRAM (PC-DDR3L, PC-DDR3L-RS).
- A power engine plug-in (PEP) that coordinates the low-power state of host controllers on the SoC with SoC-wide power states. The PEP is a small, lightweight driver that abstracts the SoC-specific power dependencies. All modern connected standby platforms must include a PEP that, at minimum, communicates to

Windows when the SoC is ready to enter the lowest-power idle mode. For Intel based platforms, the PEP is already present as an inbox driver where SoC specific information is conveyed directly through ACPI FW tables.

Communications and networking devices

The networking device(s) in a modern connected standby-capable platform are responsible for maintaining connectivity to the cloud while the SoC remains in a low-power idle mode. This capability is achieved by offloading basic network maintenance to the networking device.

The network devices in a modern connected standby-capable platform must be capable of protocol offloads. Specifically, the network device must be capable of offloading Address Resolution Protocol (ARP), Name Solicitation (NS), and several other Wi-Fi-specific protocols. To offload protocol processing, the small microcontroller on the networking device responds to network requests while the SoC remains in a low-power idle mode, saving battery power during sleep.

The network devices in a modern connected standby-capable platform must also be capable of detecting important incoming network packets and waking the SoC if necessary. The ability to detect these packets is called wake-on-LAN (WoL) patterns. With WoL patterns, the network device wakes the SoC or core silicon only when an important network packet is detected, which allows the SoC to otherwise remain in a low-power idle mode. The list of important packets to detect is provided to the networking device by Windows and correspond to the system services or apps on the lock screen.

For example, Windows always asks the network adapter to listen for incoming packets from the Windows Notification Service (WNS). Apps that are pinned to the lock screen can also request that the network device listen for app-specific packets for real-time communications, such as Skype.

For more information about protocol offloads, see Protocol Offloads for NDIS Power Management. For more information about WoL patterns, see WOL Patterns for NDIS Power Management.

System designers who build modern connected standby-capable PCs are highly encouraged to build a deep working relationship with their networking hardware vendors.

Platform requirements for modern standby

To support modern standby, a PC platform must meet the technical requirements summarized in the following table.

Topics	Description	Who's responsible?
The system ACPI firmware must set the ACPI_S0_LOW_POWER_IDLE FADT flag.	Indicates that the hardware platform supports the low-power idle mode for modern standby. **Note**: The FADT bit takes precedence over an S3 object.	System firmware developer (Core silicon or SoC must be capable of low-power idle.)
For non-Intel based platforms, the core silicon or SoC manufacturer must provide a power engine plug-in (PEP).	The PEP coordinates device state and processor idle state dependencies. A minimal PEP is required to communicate to Windows when the device power state dependencies have been achieved for the lowest SoC idle power mode.	Core silicon or SoC provider (Windows 8.1 and later includes the PEP for Intel-based platforms.)
Modern standby PCs supporting Win32 apps must also support Hibernate.	Hibernate is used to save the state of desktop/Win32 applications when critical-low battery capacity is reached.	System firmware developer
For modern standby systems, networking devices that are expected to be able to remain connected must be compatible with NDIS 6.3 (specifically WoL patterns, protocol offloads, and D0 packet coalescing).	Enables the SoC to enter low-power modes while the networking device maintains connectivity.	System designer (OEM/ODM)
Modern standby systems with soldered-in dGPU(s) or support for dGPU plug-in cards must follow Microsoft's guidelines for dGPU support.	Enables the dGPU to enter D3Cold to preserve battery life and to support VRAM self-refresh for faster resume latency.	System firmware developer and system designer (OEM/ODM)
Modern standby systems that support hibernate must implement the ACPI Time and Alarm Device (TAD) with separate AC and DC timers and support for wake on AC reattach.	Enables timers to wake the platform from hibernate depending on the power source (AC or DC) and enables expired AC timers to fire on reattach to AC power.	System firmware developer

Topics	Description	Who's responsible?
Modern standby systems must implement the Battery Trip Point (_BTP) method in ACPI.	Enables the platform to detect changes in battery percentage while in modern standby. This enables features such as adaptive hibernate to work correctly.	System firmware developer
Storage devices in Modern Standby systems should support D3 if possible.	If the platform supports D3, D3 should be enabled for the storage device(s) as described here.	System firmware developer and system designer (OEM/ODM)

Part selection

Article • 11/23/2020

If a system reports itself as supporting S0 low power idle, but does not support the networking or storage requirements to support a modern standby state, the system will automatically be placed in a disconnected state.

Start from a Modern Standby-capable chipset or Modern Standby reference platform

Starting from an existing Modern Standby system design or a reference design from your silicon partner designed for S0 low power idle is highly recommended.

> **Note**
>
> You should not try to convert existing S3 platforms to Modern Standby, as significant development investments are required to ensure proper behavior and a good power floor. This risk must be considered by OEMs, as there was no validation done by Microsoft and silicon partners to validate this design approach.

To support Modern Standby, a PC platform must meet the technical requirements summarized in Platform requirements for modern standby.

Modern Standby component selection is important. The following low power support is highly recommended for achieving a good power floor:

- USB
  - USB 2.0 - LPM support for L2
  - USB 3.0 - LPM support for U2
  - More information on USB LPM
  - Software support for legacy USB EHCI based controllers is under maintenance mode only. New modern standby capable platforms must use USB XHCI instead
- PCIe (WLAN and storage) – support for L1.substate
- For SATA SSD devices use of Slumber and Device Sleep

For more detailed guidance on platform design for systems that support S0 low power idle, contact your silicon provider.

D3 device power state support

Devices going into low power states (when not in use) is an important part of maintaining a low power floor. The same guidance for Windows 8.1 Connected Standby (for devices moving to D3 hot and D3 cold as appropriate) also applies to Modern Standby systems. Device class guidance is available on MSDN.

Storage selection

SSD storage

As in the past, SATA SSDs need to support DEVSLP in order to improve battery life in Modern Standby. Further, AHCI PCIe SSDs should ideally also expose support for DEVSLP (device sleep or SATA DEVSLP), which would likely be mapped to a L1.2 PCI sub-state by the firmware. In this case, the device should not transition between power states autonomously, so the host can control the state transitions. If the AHCI PCIe SSD does not support DEVSLP, you need to ensure the SSD can enter low-power modes on its own .

Akin to AHCI PCIe SSDs, NVMe SSDs need to provide the host with a non-operational power state that is comparable to DEVSLP (<5mW draw, <100ms exit latency) in order to allow the host to perform appropriate transitions into Modern Standby. Should the NVMe SSD not expose such a non-operational power state, autonomous power state transitions (APST) is the only other option to enter Modern Standby successfully.

Note that in the absence of DEVSLP or a comparable NVMe non-operational power state, the host can make no guarantees on the device's power draw. In this case, if you observe non-optimal power consumption by the device/system, you will have to work with your device vendor to determine the cause.

Rotational storage

Storage solutions combining flash and rotating media have shown to be generally capable to keep important data in the flash, thus allowing for quick resume times and a relatively low power profile. Storage solutions using only rotational media are also feasible, though they may result in increased power consumption and higher exit latency.

Rotating media contains many moving parts that wear out over time. When frequently power cycling a drive, the wear can be considerably increased, since the read/write heads have to move from the platter to their parked position and back.

The number of such load/unload cycles is directly influenced by the following factors:

Item	Factor	Description
1	Cache Size	How large is the non-volatile cache that can absorb I/O requests before the platter has to be accessed? A larger cache means less platter accesses.
2	Firmware Timers	Firmware Timers: At which point of idleness does the firmware automatically unload the read/write heads?
3	Cache Manager Efficiency	How well can the cache manager "predict" which data will be needed in Modern Standby and thus pre-load it into the NAND cache?
4	I/O Load & Pattern	How active is the user and how many apps are running in Modern Standby causing I/O? How frequent is this I/O? What is the user's working set?

From the above factors, #3 and #4 are very difficult or almost impossible to control. Cache Size (#1) and Firmware Timers (#2) are directly in the OEM's control.

Modern Standby causes a system to be powered down more aggressively to meet system power requirements. Without any safeguards in place, this could cause excessive wear on rotational drives. However, Windows 10 attempts to balance power savings with device reliability through a mechanism called adaptive D3 idle timeout. With adaptive D3 idle timeout, the system detects excessive power cycling and reduces it by increasing the D3 idle timeout, thus keeping the hard drive in a D0 state for a longer period. This mechanism prevents excess wear and tear to the hard drive that may reduce its long-term reliability and violate warranty claims. However, it also increases a device's power consumption.

The following points summarize storage guidance for Modern Standby systems:

- For the best experience, use pure flash devices (SSDs) and take full advantage of connectivity while in Modern Standby.
- If you use rotational or hybrid storage in some form, make sure (through your own testing and validation) that the Load/Unload cycles stay in reasonable bounds. This can be positively affected by:
  - Deploying hybrid solutions with at least 12GB useable NAND to absorb while in Modern Standby.
  - Setting firmware load/unload timers at 45 s or more (<10s is currently common) to ensure that the frequency of load/unload cycles does not exceed warranty claims.
- If you use rotational or hybrid storage, excessive load/unload cycles will cause the D3 idle timeout to increase. This results in a device staying in an active power state for longer, leading to increased power consumption.

- Expect higher exit latency upon system resume from Modern Standby with rotational storage than with SSDs.

Battery selection

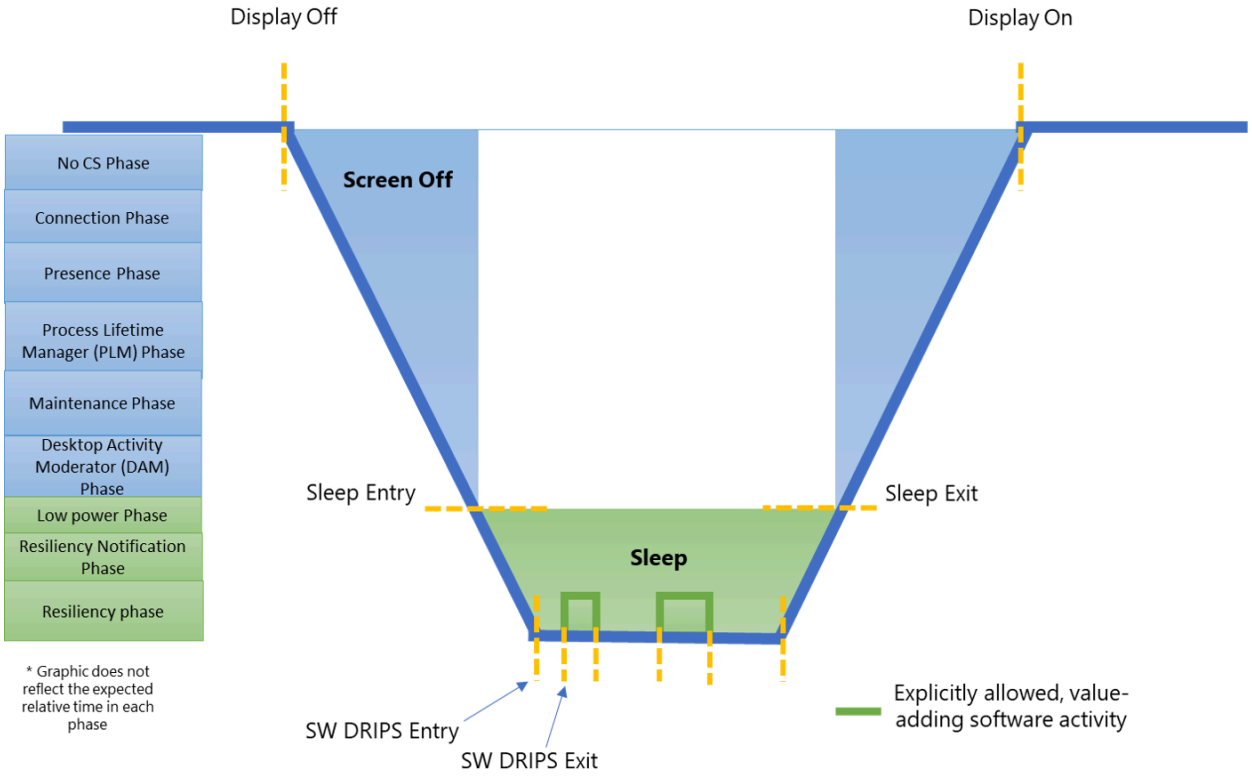
Battery life targets vary by form factor and price. Component selection and the associated power floor also impact how long a given charge will last.

Modern Standby firmware notifications

Article • 05/23/2024

In line with the Modern Standby system states outlined, Microsoft provides notifications to indicate when the system transitions into and out of these states. It is important to note that these are firmware notifications rather than OS-level notifications.

These notifications are a set of Device Specific Method (_DSM) functions that can be used by the operating system to interact with system firmware for power and functionality optimizations around key system state transitions. When designing a Modern Standby-capable system, engineers must consider how and under what circumstances certain system behaviors should be implemented. One important factor in these decisions is the system's power state—the designer may want to take certain action when the display turns off, when the system enters sleep, etc. This section seeks to provide an overview of the notifications exposed for power state transitions on Modern Standby systems. Microsoft exposes notifications for three main transitions upon entry and exit. These transitions and corresponding notifications are shown in the following diagram:



This section assumes that the reader is familiar with interfaces defined by the ACPI specification. Please refer to this page for information regarding the latest ACPI specification. Note that the following _DSM is platform agnostic, and should not be confused with the Intel-specific _DSM described here. For this _DSM, identified by its UUID, Functions 0, 3-6 are supported starting with Windows 10 version 1903, and

Functions 7 & 8 have been added in Windows 10 version 2004. Function 9 has been added in Windows 11 version 22H2.

Expand table

_DSM Function	Supported Since
Function 0, 3-6	Windows 10 version 1903
Function 7,8	Windows 10 2004
Function 9	Windows 11 22H2

Enumerate Functions (Function 0)

In order for the operating system to interact with the platform, an ACPI device must be exposed through the Namespace. **This Device must include a _CID object containing EISAID("PNP0D80").** This device's scope must contain the following _DSM definition indicating which _DSMs the device supports.

UUID	Revision	Function	Description
11E00D56-CE64-47ce-837B-1F898F9AA461	0	0	Enumerate Functions
	0	3	Display Off Notification
	0	4	Display On Notification
	0	5	Lowest Power State Entry Notification
	0	6	Lowest Power State Exit Notification
	0	7	Modern Standby Entry Notification
	0	8	Modern Standby Exit Notification

In order to indicate support for functions 3 through 8 listed above, the Enumerate Functions function (function 0) should return **Buffer(){0xFF, 0x01}** . Please refer to section 9.1.1 of the ACPI spec for more information.

Display Off Notification (Function 3)

This _DSM Function will be invoked when the operating system has entered a state where all displays—local and remote, if any—have been turned off. This could occur based on some user action, e.g. a button press or lid close event, or expiration of some display power down timer. If the OS supports Display Off Notifications by invocation of this function, it must also invoke the Display On Notifications described in the next section.

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 3
- Arg3: Unused

Return:

- None

Display On Notification (Function 4)

This _DSM Function must be invoked if a Display Off Notification has occurred and any display—local or remote—is returned to an active state. If the OS supports Display On Notifications by invocation of this function, it must also invoke the Display Off Notifications described in the previous section.

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 4
- Arg3: Unused

Return:

- None

Lowest Power State Entry Notification (Function 5)

This _DSM function is invoked when all software activity has ceased (no activators are held) allowing the system to enter its lowest power state. The system may enter and exit this state several times during a Modern Standby session to service explicitly allowed, user-desired behaviors. For more information about how this is defined and differentiated from Modern Standby Sleep entry (Function 7), please see the diagram above. If the OS supports this Lowest Platform Power State Entry Notification, it must

also invoke the corresponding exit notification described in the next section (Function 6).

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 5
- Arg3: Unused

Return:

- None

Lowest Power State Exit Notification (Function 6)

This _DSM function may be invoked whenever the system wakes from its lowest power state to run software activity (an activator is taken). The system may enter and exit this several times during a Modern Standby session to service explicitly allowed, user-desired behaviors. For more information about how this is defined and differentiated from Modern Standby Sleep entry (Function 7), please see the diagram above. If the OS supports Lowest Power State Exit Notifications by invocation of this function, it must also invoke the Lowest Power State Entry Notifications described in the previous section.

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 6
- Arg3: Unused

Return:

- None

Sleep Entry Notification (Function 7)

This _DSM function is invoked when the system enters the Sleep state of a modern standby session. This is defined as entry into the low power phase, when all ancillary system quiescing behavior has completed, and only explicitly allowed, user-desired, value-adding software activity can run. Microsoft recommends keyboard backlight, power button LED and fan turn off at this notification, as it indicates that larger workloads should be completed and the system has officially entered its long-term Sleep state. If the OS supports Modern Sleep entry notifications by invocation of this function, it must also invoke the Modern Sleep exit notifications described in the next section.

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 7
- Arg3: Unused

Return:

- None

Sleep Exit Notification (Function 8)

This _DSM function is invoked when the system exits from the Sleep state of a modern standby session. This is defined as exit from the low power phase, where the system is no longer in the explicit sleep state and may now allow larger workloads to run. Microsoft recommends keyboard backlight, power button LED and fan behavior can restore at this notification, as it indicates that larger workloads can now run as the system has officially exited its long-term Sleep state. If the OS supports Modern Sleep exit notifications by invocation of this function, it must also invoke the Modern Sleep entry notifications described in the previous section.

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 8
- Arg3: Unused

Return:

- None

Turn On Display Notification (Function 9)

This _DSM is invoked when the system intends to turn on the display when exiting Modern Standby. For example, if any OEM design were intentionally put device to lower Power Limit (PLx) when after sleep entry (Function 7), but that specific design be complained resume time longer than other devices. That Function 9 is a signal to help system firmware earlier release PL to improve resume time from Modern Standby when ONLY intends to turn on display. (Functions 9 have been added in Windows 11 version 22H2.)

Arguments:

- Arg0: UUID: 11E00D56-CE64-47ce-837B-1F898F9AA461
- Arg1: Revision ID: 0
- Arg2: Function Index: 9
- Arg3: Unused

Return:

- None

Device-specific power management for modern standby

Article • 12/12/2019

To achieve long battery life in modern standby, a platform must be able to operate at a very low hardware power floor. The term power floor describes the hardware power state in which all devices are idle and inactive, and power consumption is dominated by hardware static leakage. If properly designed, the platform typically spends well over 90 percent of a modern standby session operating at the platform's power floor.

To reliably operate at the power floor requires each device outside the System on a Chip (SoC) to enter very low standby power state when it is unused. The power management methodology and system configuration used to achieve power-floor operation is device-specific. Power to a device might be controlled autonomously by the device, based on commands from software drivers. Or, power to the device might be managed by ACPI firmware that controls power management hardware external to the device.

This section describes the supported power management configurations for devices outside of the SoC (or core silicon) in a Windows platform that implements the modern standby power model. For each class of device, the supported hardware and software power management configurations are described. The driver developer for a device should work closely with the device vendor and system integrator to review platform schematics, device integration, and ACPI firmware.

Note This documentation describes the power management configurations that Windows supports for device classes that are commonly used in modern standby platforms. This documentation does not introduce Windows Certification requirements or explicitly discuss such requirements.

In this section

Topic	Description
Audio subsystem power management for modern standby platforms	Every Windows PC has an audio subsystem that enables the user to listen to and record high-quality sound in real-time. A hardware platform that supports the modern standby power model is typically built around a SoC integrated circuit that features built-in, low-power audio processing units.

Topic	Description
Bluetooth power management for modern standby platforms	A Bluetooth radio device enables short-range RF communication between a PC and an input device, an audio device, or other Bluetooth-attached user peripheral. In a modern standby PC, the driver for a Bluetooth radio should manage this device's power states according to the guidelines presented in this article.
Camera power management for modern standby platforms	Outlines various aspects of camera power management for modern standby platforms.
GNSS receiver power management for modern standby platforms	This topic discusses Global Navigation Satellite System (GNSS) power management for modern standby-capable platforms.
Networking power management for Modern Standby platforms	Outlines networking device functionality and discusses WiFi and MBB power management for modern standby platforms.
Near-field proximity (NFP) power management for modern standby platforms	Describes near-field proximity (NFP) power management requirements for modern standby platforms.
Sensors power management for modern standby platforms	This article explains how to implement power management for sensor devices. In addition, power management of the optional sensor microcontroller (also called the sensor fusion hub or the sensor MCU) and aggregated sensor devices is discussed.

Audio subsystem power management for modern standby platforms

Article • 08/24/2021

Every Windows PC has an audio subsystem that enables the user to listen to and record high-quality sound in real-time. A hardware platform that supports the connected standby power model is typically built around a System on a Chip (SoC) integrated circuit that features built-in, low-power audio processing units.

The audio processing units offload audio processing from the main processor (or processors) on the SoC. Because these units can process audio data without using the main processor, the user can continue to listen to audio even after the main processor enters a low-power state to conserve battery power.
https://learn-video.azurefd.net/vod/player?id=205c28cd-578b-42f5-85d1-f9233aa6392e&locale=en-us&embedUrl=%2Fwindows-hardware%2Fdesign%2Fdevice-experiences%2Faudio-subsystem-power-management-for-modern-standby-platforms

This video shows how to use Windows Performance Analyzer (WPA) to verify that a computer enters the low-power state during screen-off audio playback (also known as low-power audio, or LPA).

The following article discusses audio subsystem power management for connected standby platforms. In the following discussion, the term *on-SoC* component describes a component that is integrated into the SoC chip. An *off-SoC* component is external to the SoC chip.

Audio subsystem overview

In addition to the SoC function blocks that do offloaded audio processing, each connected standby platform includes an off-SoC component, called a *codec*, that does the following:

- Translates decoded digital streams into analog sounds.
- Drives built-in speakers.
- Drives externally attached analog headphones.

Like with a camera subsystem, the audio subsystem features both on-SoC and off-SoC components. However, Windows expects a single audio driver to manage both the on-SoC audio processing engines and the off-SoC codec. This single audio driver is responsible for managing both the components that are integrated into the SoC and the

off-SoC components that can be selected by the system integrator. Therefore, the system integrator should work closely with the SoC silicon vendor on audio subsystem integration and power management.

The audio hardware vendor must implement the audio driver as a Port Class (Portcls) miniport driver. The audio driver works in conjunction with the Portcls system driver, Portcls.sys, which is an inbox component of Windows.

Compared to other device classes, the audio subsystem is unique in the way that it does power management when the platform is in the connected standby power state (that is, when the screen is turned off). When the platform is connected standby, the system can generate audio sounds to notify the user of events (for example, the arrival of a new email) in real time. Additionally, the user can turn off the system display and then continue to listen to audio being played by an application. These capabilities cannot be achieved by a simple power management strategy in which the audio subsystem must be turned off when the system is in connected standby. Instead, power management of the audio subsystem must be performed on a run-time idle basis (so that it turns on only when required) at all times except when the system is in the ACPI Shutdown (S5) state.

The audio driver performs run-time idle power management in close cooperation with the Windows audio infrastructure and the PortCls system driver. PortCls monitors any accesses (such as I/O and property accesses) of the audio device and resets the idle timer on each access. If the idle timer expires, PortCls transitions the audio device (with help from the audio driver) to a low-power sleep (D3) state. PortCls returns the audio device to the active (D0) state in the event of new access activity.

PortCls also registers with the Windows power framework (PoFx) so that the system power engine plug-in (PEP) can be notified of audio device power state changes. These notifications allow the PEP to know when it can safely turn off clocks and power rails that might be shared between the audio processing units and other SoC function blocks.

We recommend that when the audio subsystem is not being used, it should be in a sleep state in which a total of less than one milliwatt is consumed by the audio subsystem. This total includes the power consumed by the audio processing units, the off-SoC codec, and any additional audio circuitry (for example, amplifiers for speakers and headphones).

Audio subsystem hardware topology

The audio subsystem is comprised of multiple on-SoC and off-SoC components, but is presented to Windows as a single device in the ACPI namespace.

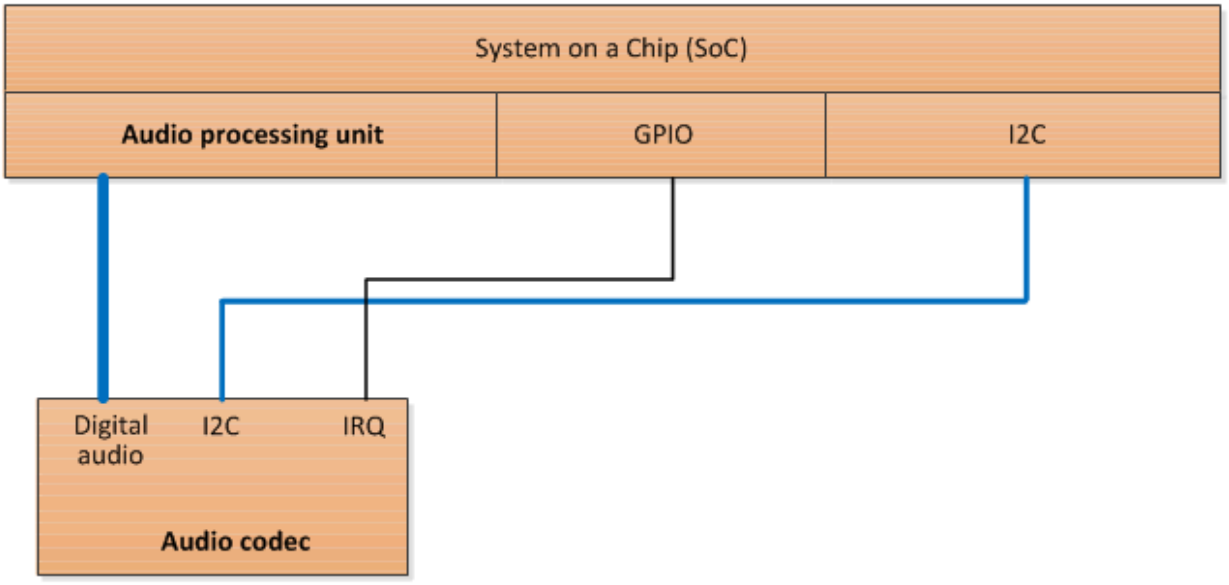
The audio processing units are located on the SoC. SoCs from different vendors have audio processing units that vary in their capabilities, power consumption, and performance. The audio processing units perform audio offloading—they process audio streams (for example, by mixing and applying audio effects) without using the main processor. For audio playback that is not latency sensitive, offloading audio from the main processor is preferred because the audio processing units use less power than the main processor.

For more information about offloaded audio, see Hardware-Offloaded Audio Processing.

The system also includes an off-SoC audio codec that converts the digital audio stream to analog output to drive built-in speakers or external headphones. The codec might include integrated analog amplifiers for speakers and headphones. Or, discrete amplifiers can be used instead. A typical codec has the following connections to the on-SoC audio processing unit:

- A digital audio interface (I2S or similar serial bus).
- A control interface (typically I2C or similar serial bus).
- One or more GPIO pins to control power management circuitry and to interrupt the SoC when the codec state changes.

These connections are shown in the following block diagram.



From the point of view of Windows, the audio processing unit and the audio codec together comprise the audio device. The audio device must be enumerated in the ACPI namespace as a single device object.

Although the audio subsystem should be exposed to Windows through a single audio driver, the SoC vendor might, as an option, adopt a driver extension model in which the audio driver is decomposed into two or more separate drivers. For example, the control software that directly manages the audio codec might be placed in a codec driver that is separate from the main audio driver. The main audio driver then indirectly manages the codec by communicating with the codec driver. The details of this driver extension model are outside the scope of this document and are proprietary to the SoC vendor's audio driver. The system integrator should work directly with the SoC silicon vendor to implement such proprietary features in the audio subsystem.

Power management modes

The audio subsystem must support the following two power management modes:

- An *active* mode in which audio is actively being streamed for the user.
- A low-power *sleep* mode in which the audio processing unit is turned off, the off-SoC codec is placed in a low-power mode, and the combined audio subsystem components consume less than one milliwatt.

The following table describes these two power modes.

Expand table

Mode	Description	Device power state (Dx)	Average power consumption	Exit latency to active	Transition mechanism
Active (streaming)	The audio processing units are actively streaming audio and the codec is providing analog or digital audio to an audio endpoint such as headphones, built-in speakers, or a remote	D0	<= 100 milliwatts (audio processing + codec)	N/A	Transition to D0 initiated by Portcls. Occurs when an application or system service initiates audio streaming.

Mode	Description	Device power state (Dx)	Average power consumption	Exit latency to active	Transition mechanism
	HDMI output device.				
Sleep	The audio processing units are not streaming audio and the codec is not operational except for standby power sufficient to detect jack insertion or removal.	D3	<= 1 milliwatt (Recommended.)	<= 35 milliseconds or <= 300 milliseconds, depending on system scenario. (Required.)	Transition to D3 initiated by Portcls. Occurs when all applications finish audio streaming and the driver-provided or system-provided idle time-out expires.

In some SoC designs, the audio processing units are multifunction blocks shared with video decoding and graphics processing. With these designs, there may be scenarios in which the audio processing units are powered on when audio is not actively streaming.

Software power management mechanisms

The primary software power management mechanism for the audio subsystem is run-time idle detection that is built into PortCls. Run-time idle detection allows PortCls to observe the application audio streaming activity to determine when to switch the audio device between the active and sleep power modes. PortCls also enables a proprietary extension mechanism between the audio driver and the SoC vendor-provided power engine plug-in (PEP) to manage the performance state of the audio processing units.

Run-time idle detection

The components in the audio subsystem enter the low-power sleep mode after the audio subsystem is idle for some specified time-out interval.

The audio driver that is provided by the SoC vendor must register the following two default idle time-out settings:

- PerformanceIdleTime – Use this time-out interval when the hardware platform is plugged into AC power.
- ConservationIdleTime – Use this time-out interval when the platform is running on battery power.

The idle time-out settings are stored in registry entries that are located under the audio driver's PowerSettings registry key. For more information, see Audio Device Class Inactivity Timer Implementation.

The following .inf directives must be used to set a PerformanceIdleTime time-out of one second and a ConservationIdleTime time-out of one second:

```
[MyAudioDevice.AddReg]
HKR,PowerSettings,ConservationIdleTime,1,01,00,00,00
HKR,PowerSettings,PerformanceIdleTime,1,01,00,00,00
HKR,PowerSettings,IdlePowerState,1,03,00,00,00
```

PortCls collaborates with the Windows kernel power manager to automatically switch between the PerformanceIdleTime and ConservationIdleTime time-out values as the platform transitions between AC power and battery power.

When the system is in connected standby (that is, with the screen turned off) and audio playback is not initiated, PortCls always uses an idle time-out of one second, regardless of the time-out setting that the adapter driver specifies in its .inf file.

The audio driver provided by the SoC vendor must also register an IdlePowerState setting to specify the power state to transition to when the idle time-out expires. On all connected standby platforms, audio drivers must register D3 as the power state to enter when an idle time-out occurs. To specify the D3 state, the **AddReg** directive in the preceding example sets the IdlePowerState value to 03.

When the idle time-out expires, PortCls calls the audio driver's **IAdapterPowerManagement3::PowerChangeState3** method to tell the driver to prepare the audio device to enter the low-power sleep mode (*NewPowerState* = **PowerDeviceD3**). The audio driver must save context for the audio processing unit and place the codec into a low-power sleep mode that consumes less than one milliwatt, on average. In the low-power sleep mode, the codec must continue to have sufficient power to detect audio jack insertion/removal and generate a level-triggered interrupt to the main processor on the SoC.

When audio playback is required because of application streaming, system sound generation, or auditory notification during connected standby, PortCls calls the audio

driver's **PowerChangeState3** method to tell the driver to configure the audio device to operate in the active (D0) power state (*NewPowerState* = **PowerDeviceD0**). The audio driver must restore context for the audio processing unit and re-enable the codec.

PortCls calls the audio driver's **IAdapterPowerManagement3::D3ExitLatencyChanged** method to notify the driver of a change in the maximum latency that can be tolerated for transitions from the sleep (D3) state to the active (D0) state. PortCls calls the audio driver's **D3ExitLatencyChanged** method to set the maximum latency to either 35 milliseconds or 300 milliseconds. The audio driver must respect the maximum latency tolerance and not enter a low-power state that requires a resume latency larger than the value specified by PortCls in the **D3ExitLatencyChanged** method.

Codec power management

The audio driver that is provided by the SoC vendor is also responsible for configuring and power-managing the off-SoC audio codec. The driver typically controls the audio codec through an I^2C or other simple peripheral bus (SPB) connection from the SoC. The driver must also handle interrupts from the codec device.

The audio driver must transition the codec to a low-power sleep mode when the audio subsystem enters the D3 (sleep) state.

The audio driver must transition the codec to the active power mode when the audio subsystem enters the D0 (active) state.

Windows power framework (PoFx) and the power engine plug-in (PEP)

PortCls registers with the Windowspower management framework so that the SoC-vendor-provided PEP is notified of audio device transitions between the active (D0) and sleep (D3) power modes. In many SoC designs, the clock and power rails for the audio processing units are shared with other on-SoC functional blocks. The PEP provided by the SoC vendor is aware of the SoC-specific clock and power topologies and takes the appropriate action to stop clocks or to turn off power rails associated with the audio processing unit when it is in the sleep mode.

Additionally, PortCls supports a private mechanism called *context sharing* that allows the audio driver to communicate directly with the PEP to perform fine-grained power management. For example, an audio driver can use context sharing to inform the PEP of the current audio stream content type and bit rate. The PEP uses this information to scale the clock frequency for the audio processing unit down to the minimum that is required to process the current audio stream without glitching.

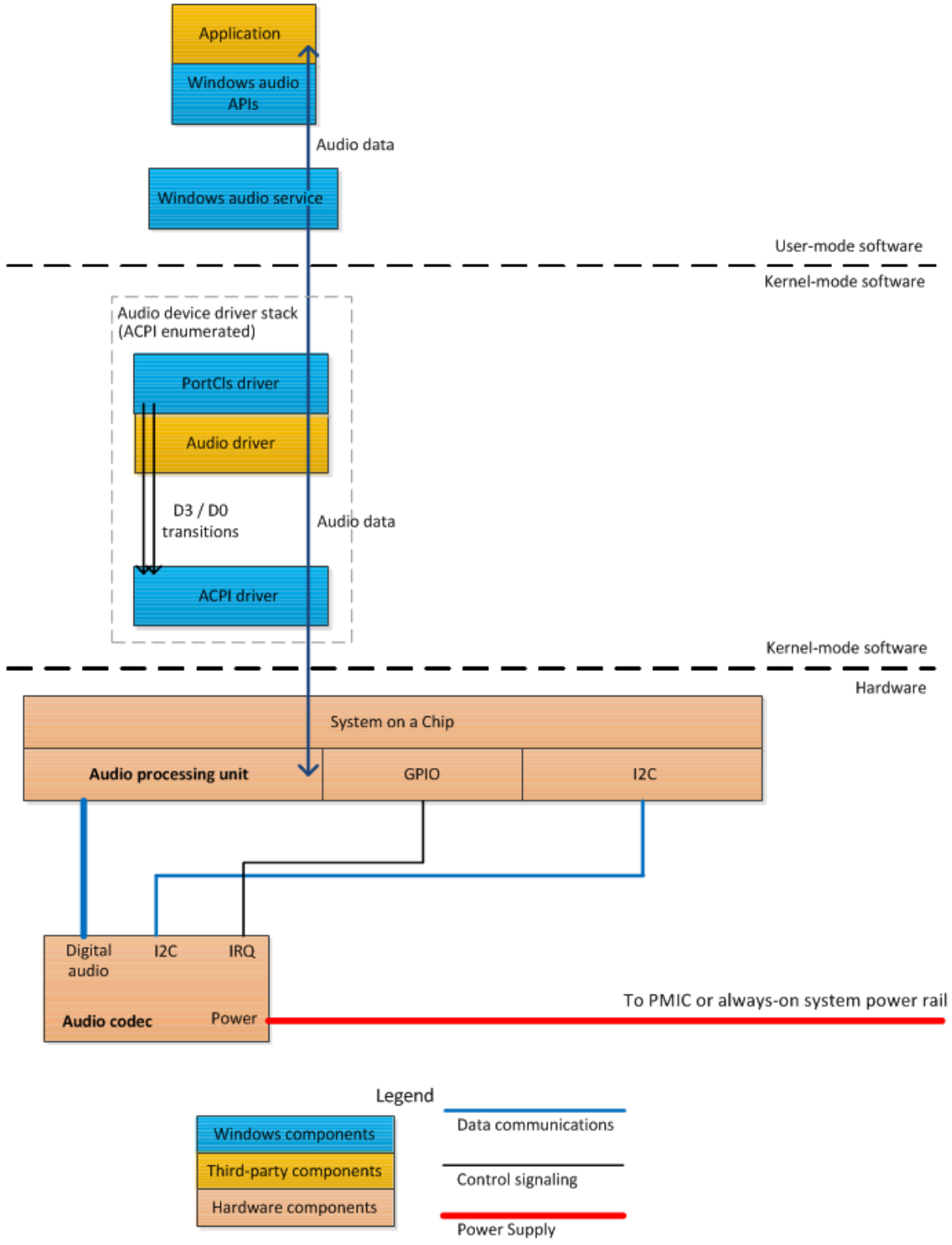
The context sharing interface is defined as a simple input/output buffer with a GUID identifier, and is similar to other extensible Windows power management interfaces. For more information about context sharing between the miniport driver and the PEP, see PortCls Private PEP Context Sharing.

Supported hardware power configuration

In connected standby platforms, Windows supports a single hardware power management configuration for the audio subsystem.

In the expected configuration, the audio processing units are located on the SoC, and the external audio codec is connected to the SoC through a SoC-compatible digital audio interface, a simple peripheral bus (SPB) such as I^2C, and one or more GPIO pins. We recommend that the audio codec and external logic consume no more than one milliwatt in the sleep power mode.

The following block diagram shows the expected hardware configuration, the **Audio device driver stack**, and the user-mode components.



The audio subsystem can have components located behind the codec that are not visible to the operating system and its drivers. For example, these components might include amplifiers for the speakers and headphones. Such components are platform-specific and can be selected by the system integrator within the requirements outlined as part of the Windows Certification program.

The system integrator must enumerate the SoC audio device in the root of the APCI namespace hierarchy. All memory, I/O, GPIO, and I^2C (or other SPB) resources required

for the audio processing unit and the external codec must be listed in the _CRS object under the device in the namespace. The system integrator and ACPI firmware developer must communicate with the audio driver developer to understand the conventions for ordering hardware resources, such as GPIO pins. For example, a driver that receives two GPIO resources distinguishes between them based on the order in which they appear in the resource list. For more information, see GPIO-Based Hardware Resources.

Although the ACPI driver (Acpi.sys) can observe the active (D0) and sleep (D3) transitions as the device power IRPs flow through the audio stack, the system integrator must not describe the audio codec as part of a power resource or use the _PS0 and _PS3 ACPI control methods to change the codec power state. In sleep mode, the codec is expected to operate at sufficiently low power that it can be left on at all times to detect jack insertion and removal.

The audio codec and any external amplifiers must be placed on a power rail that is always powered on except when the system is in the ACPI Shutdown (S5) state. GPIO pins can be used to enable or disable the amplifiers on demand. The amplifiers can be controlled by using GPIO pins from either the codec or the SoC.

A key requirement is that the codec itself remains powered at all times—even when it's in a low-power sleep mode—so that jack insertion and removal can be detected. The codec must generate an interrupt that can wake the SoC from its deepest idle state to handle headphone jack insertion and removal.

Wake concerns (headphone and microphone jack detection)

The audio subsystem must handle changes in the state of the audio output device that can occur at any time. The most common audio device state changes are the insertion of an output device into the built-in headphone jack and the removal of this device from the jack. Jack insertion and removal must also be detected for any other attached audio ports, including microphone and digital signal ports.

At all times, the audio stack must be able to detect jack insertion and removal. The interrupt line from the audio codec must be connected to a GPIO pin that is always powered and always capable of waking the SoC from its deepest idle state. Jack detection enables Windows to maintain up-to-date information about the audio input and output devices in real-time, including all times when the system is in connected standby. For example, Windows is immediately notified when the user inserts a plug into the headphones jack. In response to this notification, any future connected standby

notification alert sounds are routed to the headphones instead of to the platform's built-in speakers.

As previously described, the system firmware assigns a set of hardware resources to the audio device. These resources are described in the ACPI _CRS object, and the operating system passes a list of these resources to the audio driver. This resource list includes all GPIO interrupts that are used to detect state changes in the audio output device (for example, headphone insertion). These interrupts must be marked in the system ACPI firmware as *wake sources*. The audio driver is expected to add interrupt handlers for each of these wake interrupts. The interrupt handlers must update the state of the audio device, the audio codec, and the audio driver, as appropriate, based on which interrupt was signaled.

The ordering of resources in the _CRS object is based on a device-specific convention that is defined by the audio driver developer. For example, if the driver receives two interrupt resources, the driver distinguishes between them based on the order in which they occur in the resource list. The ACPI firmware developer must use the same ordering to describe these resources in the ACPI firmware.

Multiple hardware, firmware, and software subsystems must collaborate to make audio jack insertion and removal detection work correctly. The system integrator and audio driver developer must adhere to the following implementation guidelines:

Hardware and SoC

- The audio codec hardware must detect headphone, microphone, and other jack insertion and removal events at all times that the system is powered on, including when the system is in connected standby.
- The audio codec hardware must be able to detect jack insertion and removal while consuming very little power (less than one milliwatt average).
- The audio codec interrupt must be connected to a GPIO pin on the SoC that is capable of waking the SoC from its deepest power state.

ACPI firmware

- The audio device must be described in the ACPI namespace.
- The GPIO lines used to detect jack insertion must be described by the ACPI firmware as exclusive and wake interrupts. Use the **GpioInt** descriptor macro and set the *Shared* argument to ExclusiveAndWake.
- The audio device's hardware resources must be listed in the order that is expected by the audio driver.

Audio driver software

- The audio driver must connect an interrupt handler to the GPIO wake interrupts.
- When the audio driver handles the interrupt, it evaluates the state of the audio input/output devices and performs the appropriate actions.

Testing and validation

System integrators can use the Windows Performance Analyzer (WPA) to verify that the audio device correctly performs run-time idle power management and transitions as expected between the active (D0) and sleep (D3) states. WPA is available on the Microsoft Connect website. Please contact your Microsoft representative for assistance in obtaining WPA and the WPA Power Management extensions. The system integrator should also obtain the WPA Power Management Analysis Tools package. This package includes extensions to WPA that enable system power management analysis.

WPA relies on Event Tracing for Windows (ETW) instrumentation that is built into the Windows kernel and other Windows components, including PortCls. To use ETW tracing, a set of trace providers are enabled, and their events are recorded into a log file while a test scenario runs. When the scenario finishes, the trace providers are stopped. WPA enables post-processing and visual analysis of the log file that is generated by the scenario under test.

On a system that has WPA installed, a set of commands can be used to collect power management instrumentation to validate the power management of the audio device. The Xperf.exe tool is installed in the \%Program Files%\Windows Kits\8.0\Windows Performance Analyzer folder.

To start power management ETW tracing, open a Command Prompt window as Administrator, change to the directory that contains WPA, and run the following commands:

```
>xperf -start powertracesession -on Microsoft-Windows-Kernel-Power
>xperf -capturestate powertracesession Microsoft-Windows-Kernel-Power
```

These commands instruct Windows to enable the Microsoft-Windows-Kernel-Power ETW event provider and to capture the initial state of events from the Microsoft-Windows-Kernel-Power provider.

After the ETW tracing has started, the developer should exercise system scenarios to verify that the audio device correctly transitions between the active (D0) and sleep (D3) power modes. The developer should validate audio device in the following scenarios:

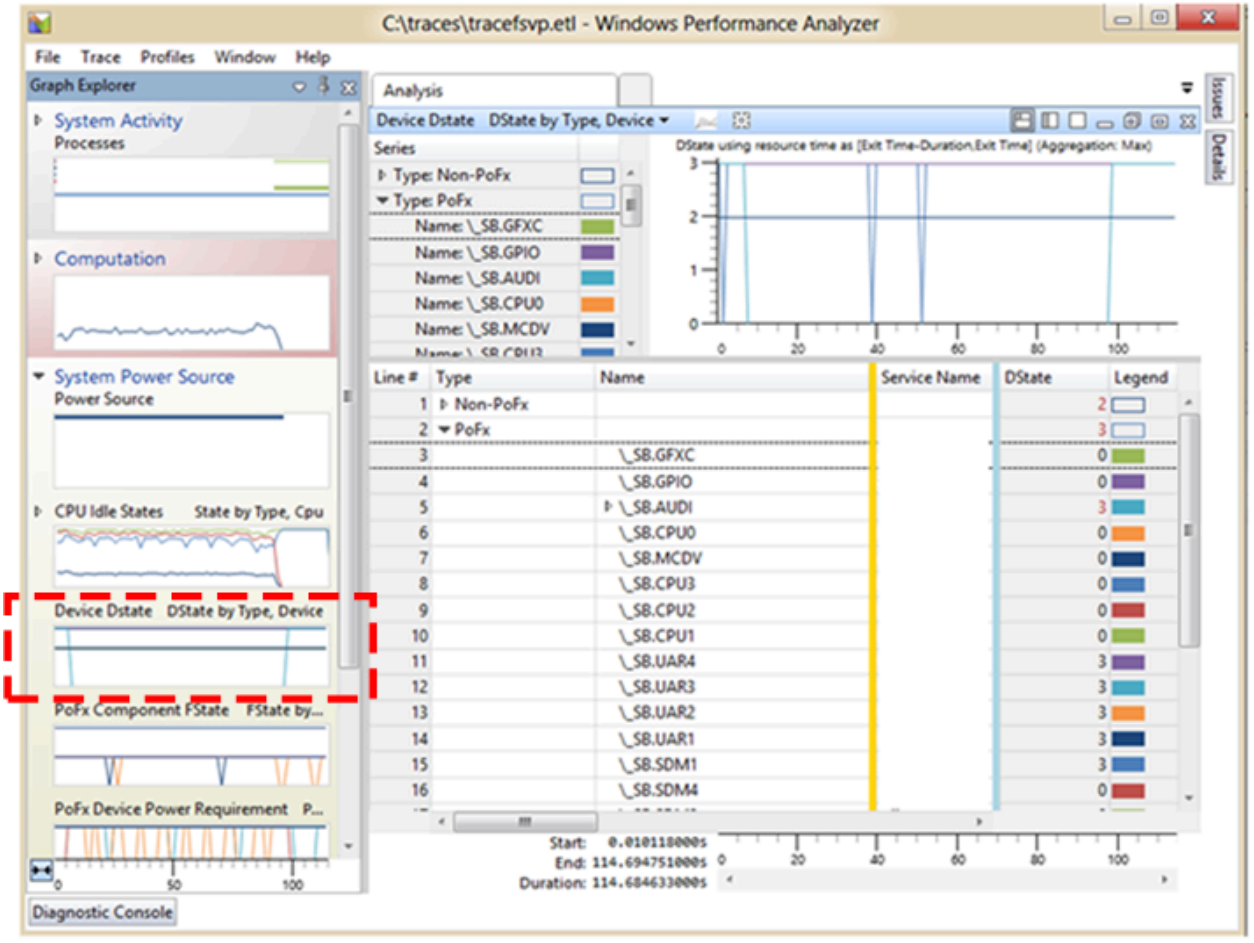
- Launch an application that transitions the audio device from the D3 state to the D0 state.
- One second after all audio applications are closed, the audio device transitions to D3 from the D0 state.
- When the system is in connected standby, the audio device remains in the D3 state.
- When an auditory notification is generated during connected standby, the audio device transitions from D3 to D0, plays audio, and then returns to D3 after one second.

After these test scenarios have completed, use the following command stop ETW trace collection:

```
>xperf -stop powertracesession -d trace.etl
```

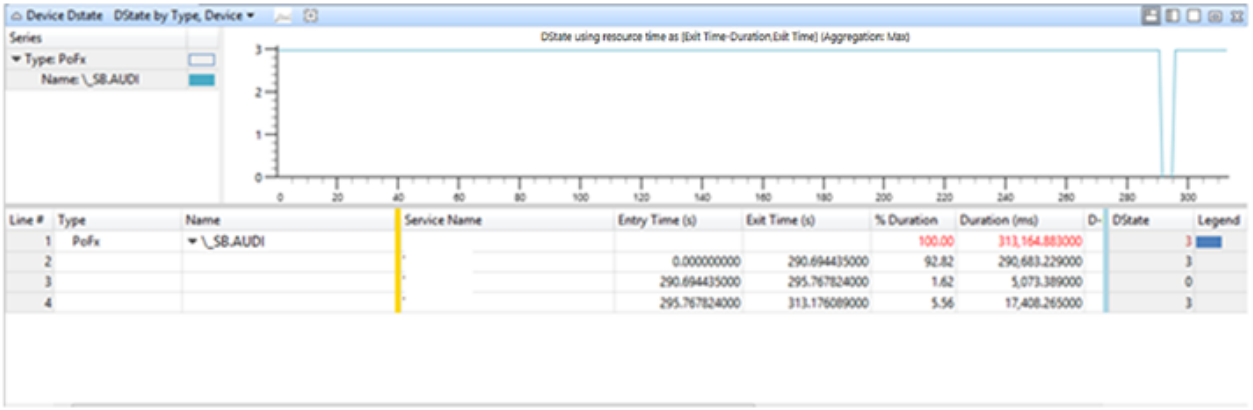
Use WPA to open the resulting Trace.etl file. To launch WPA from the command line, enter the command `Wpa.exe`.

In the WPA tool, select the **Device Dstate** graph from the **Graph Explorer** list, and the following view should appear.



In this view, a device is identified either by its ACPI name (for example, _SB.AUDI) or the device instance path (for example, ACPI\MSFT0731\4%ffff367&2). Both the ACPI name and the device instance path are listed in the summary table for the **Device Dstate** graph.

To view the D-state transitions made by the audio device, find the device name in the summary table, right-click on the name, and choose **Filter to Selection**. The resulting graph shows the D-state transitions of the audio device only, as shown in the following screenshot.



This example trace shows that the audio device was in the D3 state (indicated by coordinate **3** on vertical axis) for the entire trace duration except for one five-second period at approximately 290 seconds from the start of the trace.

Power management checklist

System integrators and SoC vendors should use the following checklist to ensure that their audio subsystem power management design is compatible with Windows 8.1.

- The system integrator should work closely with the SoC vendor to integrate audio subsystem devices.
- The audio driver developed by the SoC vendor must do the following:
  - Set run-time idle time-outs for when the system is running on AC power and on battery power. The audio driver must set both the PerformanceIdleTime value and ConservationIdleTime value to one second.
  - Set the IdlePowerState value to D3.
  - In the .inf file for the audio driver, set IdlePowerState, PerformanceIdleTime, and ConservationIdleTime to the following values:

    ```
    ```

```
[MyAudioDevice.AddReg]
HKR,PowerSettings,ConservationIdleTime,1,01,00,00,00
HKR,PowerSettings,PerformanceIdleTime,1,01,00,00,00
HKR,PowerSettings,IdlePowerState,1,03,00,00,00
```

    - The audio driver must save all audio processing unit context and place the codec into a low-power sleep mode when PortCls calls the driver's IAdapterPowerManagement3::PowerChangeState3 method with a device power state of D3.
    - The audio driver must restore all audio processing unit context and re-enable the codec when PortCls calls the driver's **PowerChangeState3** method with a device power state of D0.
    - The audio driver must not use power states that violate the D3 exit latency requirement provided by PortCls in the IAdapterPowerManagement3:D3ExitLatencyChanged method.
    - The audio driver must handle configuration and power management of the external codec.
    - The audio driver must handle level-triggered interrupts from the external codec when the codec detects jack insertion or removal.

- The SoC vendor must provide a power engine plug-in (PEP) that does the following:
    - Puts the audio processing units in a low-power state when the audio driver transitions to the sleep (D3) mode.
    - Turns on any clock and power rails needed for the audio processing units when the audio driver transitions to the active (D0) power mode.
    - Correctly scales the clock and voltage supplied to the audio processing unit according to the required level of processing activity, which depends on the audio format, content type, and bit rate.

- To develop the hardware and firmware platform for the audio subsystem, the system integrator must do the following:
    - Use a codec that, in sleep mode, consumes less than one milliwatt but can still detect jack insertion and removal events.
    - Place the codec on a system power rail that is turned on at all times, except when the system is in the ACPI Shutdown (S5) state.
    - Design the ACPI firmware to enumerate the audio subsystem as a single device in the root of the ACPI namespace hierarchy.

    - Determine the memory, interrupt, I/O, GPIO, and I^2C resource-ordering conventions expected by the audio driver and ensure that the resources are listed in the same order in the ACPI _CRS object.

- To test and validate the power management of the audio subsystem, the system integrator must do the following:
    - Verify that the audio driver transitions to the D3 power state when no applications are using the audio subsystem or generating audio for the user.
    - When in Standby from idle, verify that the audio driver stays in the active D0 power state when an application or the system is generating audio, including during audio playback when the screen is turned off.
    - When in Standby from explcit entry (power button press, verify that all audio streams are closed from the OS and the audio driver transitions to the D3 power state when an application or the system is generating audio (new in OS 24H2).
    - Verify that audio playback is performed in a glitch-free and error-free manner using the tests provided in the Windows Hardware Lab Kit (HLK).
    - Ensure that jack detection works correctly when the system is in connected standby, and that audio is correctly routed to the headphones or speakers when the user inserts the plug in the headphones jack or removes the plug from the jack.
    - Measure the power consumed by the audio processing unit, the external codec, and any additional analog amplification circuitry. Ensure that the entire audio subsystem consumes less than one milliwatt when it is in the sleep (D3) power state.

Bluetooth power management for modern standby platforms

Article • 08/24/2021

A Bluetooth radio device enables short-range RF communication between a PC and an input device, an audio device, or other Bluetooth-attached user peripheral. In a modern standby PC, the driver for a Bluetooth radio should manage this device's power states according to the guidelines presented in this article.

Bluetooth radio

In a Windows system, how the power state of the Bluetooth radio device is managed depends on the bus to which the radio is connected. On hardware platforms that support the modern standby power model, Windows supports Bluetooth radios that are connected to UARTs or to Universal Serial Bus (USB). (In theory, the Bluetooth transport bus driver model that was introduced in Windows 8 should support any underlying communication bus. Currently, Microsoft verifies modern standby compatibility only for Bluetooth radios that are connected to UARTs or USB, or are integrated into a System on a Chip (SoC)).

Just as in typical Windows driver stacks, the Bluetooth radio power policy is managed by a single power policy owner (PPO) - specifically BthPort (bthport.sys). BthPort works in conjunction with a corresponding transport-specific driver (UART or USB) to appropriately drive the radio into the desired power state. In the case of USB, this is done through USB Selective Suspend via the USB host controller. In the case of UART, an additional vendor-provided transport bus driver coordinates requests from BthPort to the Bluetooth radio device over the system-specific bus connection. To control the hardware, the driver uses a combination of in-band bus communication, coordination with the power-engine plug-in (PEP), and/or out-of-band signaling through GPIO pins.

Bluetooth radio devices typically support multiple low-power modes, some of which may be proprietary to the device itself. The Windows Bluetooth driver stack requires that a Bluetooth radio support the following three device power states:

- Active (D0)
- Sleep (D2)
- Off (D3)

Device power management for a Bluetooth radio is expected to operate in a consistent way across all system power states. The Bluetooth radio does not enter a special power

management mode when the system enters modern standby. Instead, the Bluetooth radio is transitioned in and out of the Sleep (D2) state based on idle time-outs that are managed by BthPort. To support wake from modern standby on Bluetooth-attached HID input devices, the radio stays in the Sleep (D2) state and is armed for wake. Only paired Bluetooth HID device are allowed to wake the system during modern standby. The Bluetooth radio is expected to have a very low power consumption—less than one milliwatt—in the Sleep (D2) state if no devices are connected through RF links. The power consumption can be expected to vary based on the number of associated devices, the types of those devices, and their activity patterns.

The Bluetooth radio must also support the capability to turn off the radio through the radio management user interface. This user interface control is built into Windows. After the Bluetooth radio is turned off through this user interface, the radio is transitioned to the Off (D3) power state, in which it is expected to consume nearly zero watts.

Previous versions of Windows, including Windows 8 and Windows 8 RT, require the Bluetooth device vendor to provide a radio control DLL. However, starting with Windows 8.1 and Windows RT 8.1, all Bluetooth radios in modern standby platforms should support the Bluetooth Core Specification Version 4.0. Thus, vendors are no longer required to supply a software DLL to implement the radio on/off control function. Windows now handles this function and will ignore any such DLL, even if present.

Power management modes

From a software point of view, the Bluetooth radio supports three power management modes, regardless of the bus to which the radio is connected. The Windows Bluetooth driver owns the definition of the three modes and manages the transitions into and out of these modes. The following table describes the three Bluetooth radio power modes.

Mode	Description	Device power state (Dx)	Average power consumption	Exit latency to active	Transition mechanism
Active	The Bluetooth radio is actively communicating with an associated device on behalf of an application on the operating system.	D0	Varies, specific to the scenario and associated devices.	N/A	N/A

Mode	Description	Device power state (Dx)	Average power consumption	Exit latency to active	Transition mechanism
Sleep (mostly idle with a low-rate duty cycle)	The Bluetooth radio is in a low-power state. The system has been paired with a remote Bluetooth device, but there is no connection between the two. That is, the device has been disconnected. The Bluetooth controller must be able to generate a wake signal (to the SoC if the radio is not integrated) when new data arrives from the paired device. Or, the Bluetooth radio has no associations. Or, the Bluetooth radio has an active connection that is idle (no data being sent/received) and the link is in sniff mode.	D2	< 4 milliwatts	< 100 milliseconds	Windows Bluetooth driver initiates a D2 transition using a D2 power IRP. Windows Bluetooth driver initiates a pending wait-wake IRP in the underlying transport bus driver. If the Bluetooth device is attached through USB, this state is equivalent to selective suspend. (Bluetooth selective suspend requires the device to be marked as remote wake-capable and self-powered in the USB device descriptor.)

Mode	Description	Device power state (Dx)	Average power consumption	Exit latency to active	Transition mechanism
Off	The Bluetooth radio is completely off (zero watts) or in a low-power state in which no radio state is preserved. The Bluetooth radio is not capable of generating a wake signal to the SoC in this state. The Bluetooth radio is also not able to emit or receive any radio signals—all RF components are powered off.	D3	0 watts	< 2 seconds	The Windows Bluetooth driver initiates a D3 transition using a D3 power IRP. The transport bus driver or system ACPI firmware may remove power or toggle GPIO lines to transition the Bluetooth radio hardware to the Off (D3) state.

The Bluetooth radio also supports an associated mode in which the radio transmitter can be powered down by software in response to a request from the user. When the radio is enabled for the Bluetooth device, this device is in the Active (D0) or Sleep (D2) state. When the radio for the Bluetooth device is disabled by the user, Windows stops Bluetooth activity by surprise-removing the protocol drivers and their children, and then transitioning the radio device stack to the Off (D3) state.

Software power management mechanisms

The power management of a Bluetooth radio device is driven by device Dx state transitions that are initiated by BthPort as the power policy owner (PPO). The PPO decides when the device transitions between the Active (D0), Sleep (D2), and Off (D3) states.

When the radio does not have any associated devices, Windows transitions the device to D2 and persists it in that state until the user begins the pairing process. When the radio

is associated with one or more devices, the Windows Bluetooth driver uses an idle time-out to decide when to transition the Bluetooth radio from D0 to D2. This algorithm uses the pattern of Bluetooth usage by the operating system and applications to determine when to transition the radio to the D2 state. For example, the radio transitions to D2 several seconds after the last key press on a Bluetooth keyboard if there is no other activity on the Bluetooth radio.

The Windows Bluetooth driver transitions the device to D0 in response to any of the following:

- The user begins a pairing process.
- An application requests the use of Bluetooth functionality.
- The Bluetooth radio has generated a wake request based on input from an associated device.

Unlike other devices, the Bluetooth radio follows the same power management pattern during modern standby (system display off) that it does when the system is normally operational and the display is on. That's because the Bluetooth radio is expected to be available to wake the SoC when input is received from an associated device at any time during modern standby. For example, if a user has associated a Bluetooth keyboard with a Windows computer, pressing any key on the keyboard should wake the computer from modern standby and turn on the display.

If no devices are associated with the radio, the radio is expected to be configured to consume less than one milliwatt when it's in the Sleep (D2) state.

When the Bluetooth radio is in the Off (D3) state, it is expected to consume nearly zero watts.

Driver implementation notes

If the Bluetooth radio is connected through a UART or integrated into the SoC itself, the Bluetooth device vendor is required to implement and provide a transport bus driver. The transport bus driver is responsible for the following:

- Translating Bluetooth HCI packet requests from the Windows Bluetooth driver (Bthmini.sys) to commands that are sent over the transport bus to the Bluetooth radio.
- Transitioning the Bluetooth radio device into various power management modes that map to the Active (D0), Sleep (D2), and Off (D3) device power states. The driver also implements routines that handle power management events.

- Configuring the Bluetooth radio to wake the SoC when a device generates input, and changing the state of any optional GPIO lines from the SoC to the Bluetooth radio that are used for power management.
- Enumerating and power-managing other devices (such as an FM transmitter or GPS device) that share the same bus as the Bluetooth radio. If other devices are physically connected to the shared bus, but are not exposed to the operating system, the transport bus driver must completely power down these devices.

For details about implementing a transport bus driver, see Transport Bus Driver for Bluetooth Power Control Handling Guidelines. Transport bus drivers must be written using the Windows Driver Framework (WDF). A sample driver is available at Bluetooth Serial HCI Bus Driver.

To enable Bluetooth radio power management, the transport bus driver must perform the following actions:

- Enable support for run-time idle power management and expose support for the Active (D0), Sleep (D2), and Off (D3) device power states.
- Indicate to the Windows Bluetooth driver that the Bluetooth radio device is capable of signaling wake events from the D2 state.
- Support arming the Bluetooth radio device to wake the SoC and disarming the Bluetooth device's wake signal to the SoC. This support might require handling one or more GPIO interrupts and executing wake methods within WDF.
- Change the state of any optional GPIO lines from the SoC to the Bluetooth radio device when the device transitions between the Active (D0), Sleep (D2), and Off (D3) states.

If the Bluetooth radio is integrated into the SoC itself, the transport bus driver can register with the Windows power management framework to communicate Bluetooth radio power status to a SoC-specific power-engine plug-in (PEP). This is accomplished by setting the IdleTimeoutType member of the WDF_DEVICE_POWER_POLICY_IDLE_SETTINGS structure to the value SystemManagedIdleTimeout.

If the Bluetooth radio is connected through USB, the built-in Windows USB Bluetooth driver stack must be used. The stack handles all power management operations.

Radio management

The state of the Bluetooth radio transmitter is tied directly to the device power state. The radio transmitter is expected to be on when the radio is in the Active (D0) or Sleep

(D2) power state. The radio transmitter must be turned off when the radio transitions to the Off (D3) state.

When the user turns off the Bluetooth radio, Windows terminates Bluetooth activity by canceling pending I/O operations and unloading the protocol drivers and their children. The Windows Bluetooth driver stack then issues the HCI_Reset command to the controller to reset the radio to its default state. In the default state, the controller must not be capable of transmitting or receiving any radio signals. Finally, the controller transitions to the Off (D3) state.

In response to the transition to Off (D3), the transport bus driver must power off the Bluetooth device to its lowest power state using device-specific methods. A typical implementation is to change the state of a GPIO line from the SoC to the Bluetooth radio to disable power to the Bluetooth module. An alternate implementation is to require the ACPI firmware to remove power from the Bluetooth module using _PS0 and _PS3 control methods.

When the user turns on the Bluetooth radio, Windows transitions the radio to the Active (D0) state, re-initializes the radio, and then re-enumerates child protocol drivers. When the radio transitions to Active (D0), any required GPIO lines must be toggled as part of the normal D0 sequence for the Bluetooth radio. If ACPI firmware was used to power down the radio, it must restore power using the _PS0 control method.

As part of this normal sequence, the transport bus driver must mark the device as an Internally Connected Device by setting the ContainerId of the Bluetooth radio to a specific GUID value, {00000000-0000-000-ffff-ffffffffffff}. This enables the Windows radio user interface elements to detect that the Bluetooth radio exposed by the transport bus driver is internal to the computer and not an externally attached radio for which radio control is not appropriate.

Supported hardware power configurations

The power management hardware configuration for a Bluetooth radio depends on the communication bus. Generally speaking, all Bluetooth radios are expected to have the following hardware power management features in common:

- Support for the Off (D3) state as a means to turn off the radio in response to a user request. Turning off the radio puts the Bluetooth radio in a low-power state that is nearly zero watts.
- A mechanism to enter a low-power Sleep (D2) state in which connections are persisted to associated devices, but there are no active transfers.

- A mechanism to generate a wake interrupt when an associated device has data for the SoC and the SoC is in a low-power state in which the bus to which the Bluetooth radio device is attached is not currently active.

Each of the supported buses (USB, UART, and integration into the SoC) for the Bluetooth radio device support all three basic hardware power management features in the preceding list. In addition, each Bluetooth radio can have vendor-specific or device-specific power management features, but these are outside the scope of this topic.

Bluetooth radio vendors are encouraged to implement value-add power management features in a way that is autonomous in hardware and does not require additional vendor-supplied driver software on the Windows system. Bluetooth radio vendors are also encouraged to implement their driver(s) and their power management software in a way that abstracts platform-specific differences into system ACPI firmware instead of into the device driver code or the driver .inf file. This approach allows one driver package for the Bluetooth device to be re-used in additional platforms without requiring an update to the driver source, binary, or signed installation package.

Bluetooth radio that is connected through a UART outside of the SoC

If the Bluetooth radio is connected through a UART and is physically located outside of the SoC, the Bluetooth radio vendor must provide a transport bus driver that exposes the Bluetooth radio and any other device functions (for example, an FM radio) that share the same communication path through the UART. The bus driver is also responsible for managing any GPIO resources that control the Bluetooth radio's power consumption and wake capability.

Unlike other device classes, the GPIO lines that control Bluetooth power and wake are managed directly by the transport bus driver instead of being abstracted by ACPI control methods. This control scheme is a result of the multifunction bus driver design that enumerates the Bluetooth radio and other functions that share the same UART port. In this design, the Windows ACPI driver, Acpi.sys, is not loaded in the driver stacks for Bluetooth and the FM radio, making it impossible to use ACPI control method execution as a way to respond to a device Dx state transition.

When the Bluetooth radio is connected to the UART port on the SoC, the system integrator must use a pin on the GPIO controller on the SoC to control power to the radio. In the ACPI firmware, this pin must be assigned as a GPIO I/O resource to the device object that represents the transport bus driver root device. The GPIO pin can be

connected directly to the Bluetooth radio if the radio supports turning off the device with internal power-gating.

If the Bluetooth radio supports power-gating, the power source for the Bluetooth radio can be connected to any system power rail.

If the radio does not support internal power-gating that is controlled by a GPIO pin, the system integrator must place the Bluetooth radio on a power rail that is switchable. The GPIO pin from the SoC is then connected to the power-switching hardware. In this design, ACPI control methods cannot be used to track reference counts or to aggregate the power state of multiple devices sharing the same power rail, so the Bluetooth radio must be isolated on its own switchable power rail.

The system integrator must use an additional pin on the GPIO controller on the SoC to receive wake interrupts from the Bluetooth radio. Interrupts on this pin must be capable of waking the SoC from its lowest power state. In some SoC designs, such a pin is referred to as an always-on GPIO pin because the GPIO controller can detect interrupts on this pin at all times, regardless of the power state of the SoC. The always-on capability may be limited in hardware to a specific set of GPIO pins on the SoC or may be configurable in firmware. It is critical that the system integrator review this design with the SoC vendor to ensure the Bluetooth radio's wake interrupt will cause the SoC to exit its deepest idle power state. (At all times during modern standby, the system is in S0. modern standby systems do not support S3.)

When all functions enumerated by the transport bus driver have been powered off and the ACPI-enumerated transport bus device enters D3, the always-on GPIO pin can be powered off. This occurs when the radios for all device functions enumerated by the transport bus driver have been turned off by the user.

Bluetooth radio on USB

If the Bluetooth radio is attached to the SoC or core silicon over the USB bus, the radio must be powered from a source other than the USB bus. In the USB specification, such a radio is described as self-powered, and this capability must be reported in the Bluetooth device's USB descriptors.

Similarly, the USB device hardware must advertise support for remote wake, which is the capability for the Bluetooth radio to generate in-band USB resume signaling to wake up the USB host controller. The remote-wake capability must also be advertised in the Bluetooth radio's USB descriptors.

The Bluetooth radio must support both the self-powered and remote-wake capabilities so that it can enter the Sleep (D2) state and enable selective suspend.

If the Bluetooth radio is in the Sleep (D2) state and data from an associated device is available for the host, the Bluetooth radio must generate the remote wake resume signaling to wake the host. An out-of-band resume signal using a GPIO line to the core silicon is not supported. The Bluetooth radio, including its USB connection circuitry, is expected to consume less than one milliwatt of power in the Sleep (D2) state.

Wake concerns

The Bluetooth radio is expected to be able to generate a wake interrupt when in the Sleep (D2) state. The wake interrupt must cause the SoC to power on, even if the SoC is in its lowest power state. The following table summarizes the two Bluetooth wake-signaling mechanisms.

Connection bus	Hardware signaling path	Comments and notes
UART (with vendor-supplied transport bus driver)	GPIO from the Bluetooth radio to the SoC.	The radio must be connected to a GPIO pin that can wake the SoC from its lowest power state.
USB	In-band USB resume signaling from selective suspend.	Out-of-band GPIO wake is not supported.

Testing and validation

Bluetooth device vendors are encouraged to test and validate the power management operation of the Bluetooth radio device.

The transitions between the Active (D0), Sleep (D2), and Off (D3) states can easily be observed using the Xperf tool, as described in other sections.

Bluetooth driver activities can be monitored by using ETW instrumentation that is built into Windows. The driver developer is encouraged to use Event Tracing for Windows (ETW) instrumentation to expose significant power management state changes in the driver and observe those by using the Xperf tool or the built-in Windows Event Viewer.

If the Bluetooth radio is attached through USB, the built-in Powercfg.exe utility can be used together with the /energy command-line option to validate that the radio is entering the Sleep (D2) state and is suspended. To use the Powercfg.exe utility:

- Open a Command Prompt window as Administrator.
- Change to the root directory of the drive (**cd **).
- Enter the command **powercfg.exe /energy**.
- Wait for the default 60 seconds.
- The Powercfg.exe utility will output the number of errors and warning conditions on the system, as shown in the following screenshot.
- After the tool writes the summary information to the command prompt window, it generates an HTML file that is named Energy-report.html. Open the file and look for error or warning conditions from the USB Bluetooth device. The following example summary reports that a USB Bluetooth device has not entered the Sleep (D2) state when it is idle.

Bluetooth device vendors who provide additional third-party Bluetooth profile drivers and applications must verify that their software supports surprise removal and allows the radio management infrastructure to properly turn off the Bluetooth radio in a timely manner. These scenarios should be validated while the profile or application is in use. For example, for audio drivers, there should be Bluetooth audio streaming while the radio is turned off. Then the radio should be turned back on and the audio stream restarted to verify that it still works.

Bluetooth power management checklist

System integrators, Bluetooth radio vendors, and SoC vendors should use the following checklist to ensure that their system power management design is compatible with Windows 8 and Windows 8.1:

- Determine the communication bus for the Bluetooth radio in the system design. The Bluetooth radio is either connected over UART or attached via USB.

- Ensure that the Bluetooth radio supports a low-power sleep mode that consumes less than one milliwatt when no devices are associated.

  The power consumption of the Bluetooth radio in sleep mode can vary based on the number of associated devices that are currently present, but should generally not exceed five milliwatts at any time.

- Ensure that the Bluetooth radio supports the following basic required power management capabilities:
  - Support for the Off (D3) state as a way to allow the user to turn off the radio.
  - A mechanism to enter a low-power Sleep (D2) state where connections are persisted to associated devices, but there are no active transfers.

    - A mechanism to wake the SoC when an associated device generates data and the SoC is in a low-power state.

- If the Bluetooth radio is connected over a non-USB bus (UART or integrated into the SoC), the Bluetooth radio vendor must develop a transport bus driver. The transport bus driver must do the following:
    - Support the features and requirements detailed in Transport Bus Driver for Bluetooth Power Control Handling Guidelines.
    - Translate Bluetooth requests from the Windows Bluetooth driver (Bthmini.sys) to commands to the Bluetooth radio over the UART bus or a proprietary internal SoC bus.
    - Transition the Bluetooth radio device into various power management modes that map to the Active (D0), Sleep (D2), and Off (D3) states. The driver must also implement routines that handle device power management (Dx) IRPs.
    - Configure the Bluetooth radio to wake the SoC when a device generates input, and change the state of any optional GPIO lines that connect the SoC to the Bluetooth radio for power management purposes.
    - Enumerate other devices (for example, an FM transmitter) that might be shared on the Bluetooth radio.
    - Use the Windows Driver Framework (WDF) for driver development.
    - Be implemented based on the Bluetooth Serial HCI Bus Driver.

- If the Bluetooth radio is connected over USB, the Bluetooth radio vendor must:
    - Enable support for selective suspend in the radio.
    - Ensure the radio has the remote-wake and self-powered capabilities set in the USB device descriptor.
    - Ensure the radio (including USB components) consumes less than one milliwatt.

- Regardless of connection bus, the Bluetooth radio must do the following for an internally connected radio:
    - Ensure all RF components are turned off in response to a HCI_Reset command being sent to the radio in preparation for powering down the radio. The radio should not be capable of transmitting nor receiving any radio signals.
    - Enter its lowest power mode when set to the Off (D3) state.

- If the Bluetooth radio is connected over UART, the system integrator must connect the wake signal from the Bluetooth radio to a GPIO pin on the SoC that can wake the SoC from the lowest power state.
    - The SoC can require wake signals to be routed to a limited set of GPIO pins that are pre-configured to be always-on.
    - Or, the SoC can support configuration of a GPIO pin to an always-on pin in system firmware during boot.

- The system integrator must test and verify that the Bluetooth radio enters the Sleep (D2) state when there are no associated devices.
- The system integrator must test and verify that the Bluetooth radio enters the Sleep (D2) state when all associated devices do not have any active transfers.
- The system integrator must test and verify that the Bluetooth radio can wake the SoC from its lowest power state when the radio is in the Sleep (D2) state.
- The system integrator must test and verify that the Bluetooth radio does not generate spurious wake signals while in the Sleep (D2) state.
- The system integrator must test and verify that add-on third-party software, such as profile drivers and applications, work properly with Bluetooth radio management. The radio should be turned off and on while the third-party software is actively in use (for example, playing audio or transferring a file).

Camera power management for modern standby platforms

Article • 12/11/2019

The cameras in modern mobile platforms allow users to capture still and motion video of their surroundings, and to use video and audio to communicate with other users over the Internet. The general goal of power management for a camera device can be described simply—the camera subsystem must be powered off, consuming zero watts, unless the camera is in active use.

When the camera is actively being used to stream video to an application, the camera sensor and related components should be powered on. Windows enables the camera hardware to be turned off during modern standby by suspending any foreground Microsoft Store apps that might be streaming data from the camera device. Windows makes no provisions for the use of camera devices while the system is in modern standby, so a camera device can be used only when the display is turned on.

Camera subsystem

In terms of power management, cameras are one of the most complex device subsystems in a mobile platform. This complexity is a result of the tight coordination required between the digital and image signal processing units, which are integrated into the System on a Chip (SoC), and the camera sensor, auto-focus, and flash components, which are external to the SoC chip.

In the following discussion, the term on-SoCcomponent describes a component that is integrated into the SoC chip. An off-SoC component is external to the SoC chip.

Camera subsystem design becomes even more complex if the on-SoC image processing hardware must be multiplexed between two or more camera subsystems.

Tablet PCs are expected to have both a front-facing and a rear-facing camera, and these cameras share the same on-SoC image processing hardware. Hardware sharing implies multiplexing at both hardware and software layers. Because of this complexity, the system integrator should work closely with the SoC silicon vendor to integrate the cameras into the platform and implement camera device power management.

When the system is in modern standby (display off), the camera sensor, camera processing unit, optional auto-focuser, and flash components must have their power removed and consume zero watts. For the camera components on the SoC, the SoC

vendor must provide a driver that manages the image processing components. The image processing driver coordinates power management of the on-SoC components with the power engine plug-in (PEP).

For the off-SoC camera components, which include the sensor, auto-focuser, and flash, the system integrator must provide power rail switching capability and associated GPIO control signaling, which are controlled by the ACPI firmware.

The system integrator must also provide one or more device drivers to directly manage the sensor, auto-focuser, and flash hardware. Due to camera subsystem complexity and the dependencies between the on-SoC and off-SoC components, the camera sensor and flash drivers are typically provided by the SoC vendor. The system integrator must also provide a camera controller driver, which is a kernel-mode AVStream minidriver. The camera controller driver is responsible for exposing the camera devices to the Windows multimedia subsystem. However, we recommend that this driver not directly control any platform hardware due to complexities in the AVStream minidriver model. Instead, we recommend that the camera controller driver should rely on the other camera component drivers for access to hardware resources and device power management. That is, the camera controller driver should communicate with the driver that controls the on-SoC image processing hardware and the driver or drivers that control the off-SoC sensor and flash units.

In this section

Topic	Description
Camera hardware	Provides an overview of camera hardware topology. Windows supports a single hardware power management configuration for camera devices in modern standby platforms. In short, each camera sensor must be connected to the SoC via a MIPI-CSI link, and can optionally be connected to an I2C bus and to one or more GPIO pins. The camera sensor device, its optional flash, and any other off-SoC camera components must be placed on a power rail that can be switched on and off by ACPI firmware.

Topic	Description
Camera power management	Describes the power management modes that must be supported in the off-System on a Chip (SoC) and on-SoC components of the camera subsystem. Both the on-SoC image processing units and the off-SoC camera components are expected to consume no power (zero watts) when the system is in connected standby and the display is turned off. The primary software mechanism for power management is reference counting of the camera capture pin. Includes a checklist that system integrators, camera sensor vendors, and SoC vendors should use to ensure that their system power management design is compatible with Windows 10.

Camera hardware

Article • 12/11/2019

Topology overview

In terms of Windows driver support, the camera subsystem contains off-System on a Chip (SoC) components such as the camera sensor, an optional auto-focus unit, and flash—and possibly other associated hardware. The camera hardware also includes the on-SoC image processing units.

The on-SoC image processing hardware must be power-managed by the power engine plug-in (PEP) supplied by the SoC vendor. The image processing hardware should be enumerated as a single device in ACPI and managed by a Windows Driver Framework (WDF) driver. Enable system management of the image processing device idle time-out so that the PEP can control any clock and power rail sharing topology that is unique to the SoC. Any on-SoC image processing hardware must be powered off whenever the camera device is turned off.

Some SoC designs have a shared function block that does both camera capture and other image and graphics processing. In a platform that uses such a SoC, the PEP supplied by the SoC vendor must reference-count the use of this shared block and power it off when all clients are idle.

In some platforms, the on-SoC image processing hardware might be shared between two or more camera devices. In this case, the image processing hardware is multiplexed between the camera devices. The components of each camera device should be described independently in the ACPI namespace and should be enumerated as separate device objects to the Windows Plug and Play manager.

Platforms that have two (or more) embedded cameras are required by Windows to be able to use (that is, stream content from) both (or all) cameras at the same time with any combination of the modes and resolutions that are supported by the individual cameras. SoC vendors who cannot meet this requirement must work with Microsoft directly for guidance on implementing their drivers and system firmware.

Supported power configuration

Windows supports a single hardware power management configuration for camera devices in modern standby platforms. In short, each camera sensor must be connected to the System on a Chip (SoC) via a MIPI-CSI link, and can optionally be connected to an

I2C bus and to one or more GPIO pins. The camera sensor device, its optional flash, and any other off-SoC camera components must be placed on a power rail that can be switched on and off by ACPI firmware.

If, in addition to a MIPI-CSI link, the camera device has I2C or GPIO pins to control the camera sensor or flash device, these pins must be routed to the corresponding pins of the I2C controller or GPIO controller on the SoC. The system integrator must enumerate the I2C and GPIO resources for both the camera sensor and the flash device in a _CRS object under the camera device in the ACPI namespace.

Note The system integrator should work with the camera subsystem driver developer to determine how the camera drivers expect GPIO and I2C resources to be ordered. For example, a driver that receives two I2C resources distinguishes between them based on the order in which they appear in the resource list. Similarly, a driver that receives three GPIO resources expects these resources to be listed in a particular order. The system integrator must enumerate the I2C and GPIO resources in the same order in the _CRS object.

The camera sensor and the flash device must be placed on a power rail that can be turned on and off by ACPI control methods. We recommend using a GPIO pin from the SoC to control the power-switch hardware. The GPIO should be enumerated in a GPIO operation region so that its state can be changed by ACPI control methods. The system integrator must describe the power resource for a camera device (sensor, flash, or any other camera component) in the ACPI namespace. This resource must include an _ON method and an _OFF method to change the state of the GPIO signal routed to the power-switch hardware. Under the camera device in the ACPI namespace, the system integrator must provide a _PR0 object and a _PR3 object that refer to the power resource.

When the camera controller driver detects that all streaming pins have entered the KSSTATE_STOP state, it uses a private interface to tell the drivers that control the off-SoC camera components that capture is no longer required. In turn, these drivers call the IWDFDevice2::ResumeIdle method to tell the driver framework that their hardware is idle. In response, the driver framework initiates a transition to D3, which causes a D3 IRP to flow through the camera device driver stack. (A D3 IRP is an IRP_MJ_POWER IRP that specifies a DEVICE_POWER_STATE enumeration value of PowerDeviceD3.) The Windows ACPI driver, Acpi.sys, will observe the D3 IRP and execute the _OFF method of the power resource that is identified by the _PR3 object under the camera device in the ACPI namespace.

The last sentence in the preceding paragraph assumes that the power resource does not provide power to any devices other than the one camera device. If other devices have

references to this power resource, Acpi.sys will run the _OFF method only after all other devices that reference the power resource have transitioned to D3. For more information, see Enabling Transitions to D3cold.

Returning the camera hardware to the active power state is a similar process. When the camera controller driver detects the first stream capture pin to enter the KSSTATE_ACQUIRE state, the camera controller driver communicates with the drivers for the other on-SoC and off-SoC components that comprise the camera subsystem. In response, the driver that controls the on-SoC image processing unit calls the IWDFDevice2::StopIdle method, which informs the PEP that the image processing unit hardware should be powered on. The camera controller driver tells the drivers that control the off-SoC camera components that they should return to the active state. In turn, these drivers call StopIdle to inform the driver framework that the hardware is no longer idle, which causes a D0 IRP to flow through the camera device driver stack. (A D0 IRP is an IRP_MJ_POWER IRP that specifies a DEVICE_POWER_STATE enumeration value of PowerDeviceD0.) Acpi.sys responds to the D0 IRP by executing the _ON method of the power resource that is identified by the _PR0 object under the camera device in the ACPI namespace.

If the platform has multiple camera devices, each camera device should have its own independently switchable power rail and power resource described in the ACPI namespace. For each camera device in the ACPI namespace, the system integrator must provide a _PLD object that indicates whether the camera device is on the front or the back of the computer. If a camera device is built into the lid of a clamshell-form-factor computer and faces the user when the lid is open, this device's _PLD object should indicate that the camera is on the front of the platform. If a camera device is built into the lid of a clamshell-form-factor computer and faces away from the user when the lid is open, this device's _PLD object should indicate that the camera is on the back of the system.

Wake concerns

Camera device hardware must not support wake. Windows does not expect camera devices to be able to wake the SoC from its lowest power state during modern standby. Many cellular phones enable the SoC to wake from sleep when the user presses the camera button. The camera button is treated by Windows as a user input device whose operation is separate from and independent of the sytem integration or power management of the camera device, its sensor, and optional flash.

Camera power management

Article • 08/24/2021

Power management modes

The off-System on a Chip (SoC) components of the camera subsystem must support two power management modes. The camera components must support an active mode in which the camera device is actively streaming content to an application. In addition, the camera components must support a power-removed mode in which the camera device is turned off, power is removed, and the camera device consumes zero watts. The following table describes the active and power-removed power management modes for the camera device.

Mode	Description	Device power state (Dx)	Average power consumption	Exit latency to active	Transition mechanism
Active (streaming)	The camera device is actively streaming content to an application. The content might be full-motion, preview, or still photo capture.	Yes	Sensor, AF, and flash-specific.	N/A	Software-initiated D0 transition. (An application has initiated streaming by setting the state of a capture pin to KSSTATE_ACQUIRE.)
Power - removed	The camera device is not streaming content to any applications. No context is preserved on the camera sensor, the flash device, or the auto-focus engine.	Yes	0 watts	< 200 milliseconds to first frame (See note following table.)	Software-initiated D3 transition. (The state of all streaming pins has been set to any value other than KSSTATE_RUN.)

Note Windows expects the transition time from the active mode to the power-removed mode (the off latency) to be less than 100 milliseconds. Most power management effort is focused on reducing the transition time from power-removed mode to active mode (the on latency).

The same two power management modes, active and power-removed, must be supported by the on-SoC image processing units. The SoC vendor defines the individual components that comprise the image processing units and their power management states. We recommend that a single driver control the on-SoC image processing units, and that all the image processing units for camera capture be presented to the power engine plug-in (PEP) as a single power-managed component.

Software power management mechanisms

Both the on-System on a Chip (SoC) image processing units and the off-SoC camera components are expected to consume no power (zero watts) when the system is in connected standby and the display is turned off. The primary software mechanism for power management is reference counting of the camera capture pin. This reference count is maintained by the camera controller driver, which is an AVStream minidriver. This basic power management mechanism can be used any time the system is turned on, including times when the system display is powered on.

The camera controller driver should forward power management state transitions to the drivers that control off-SoC components such as the camera sensor, auto-focuser, and flash. In response, the drivers that control these devices should take specific action to change power states and to remove or apply power.

Recommended driver architecture

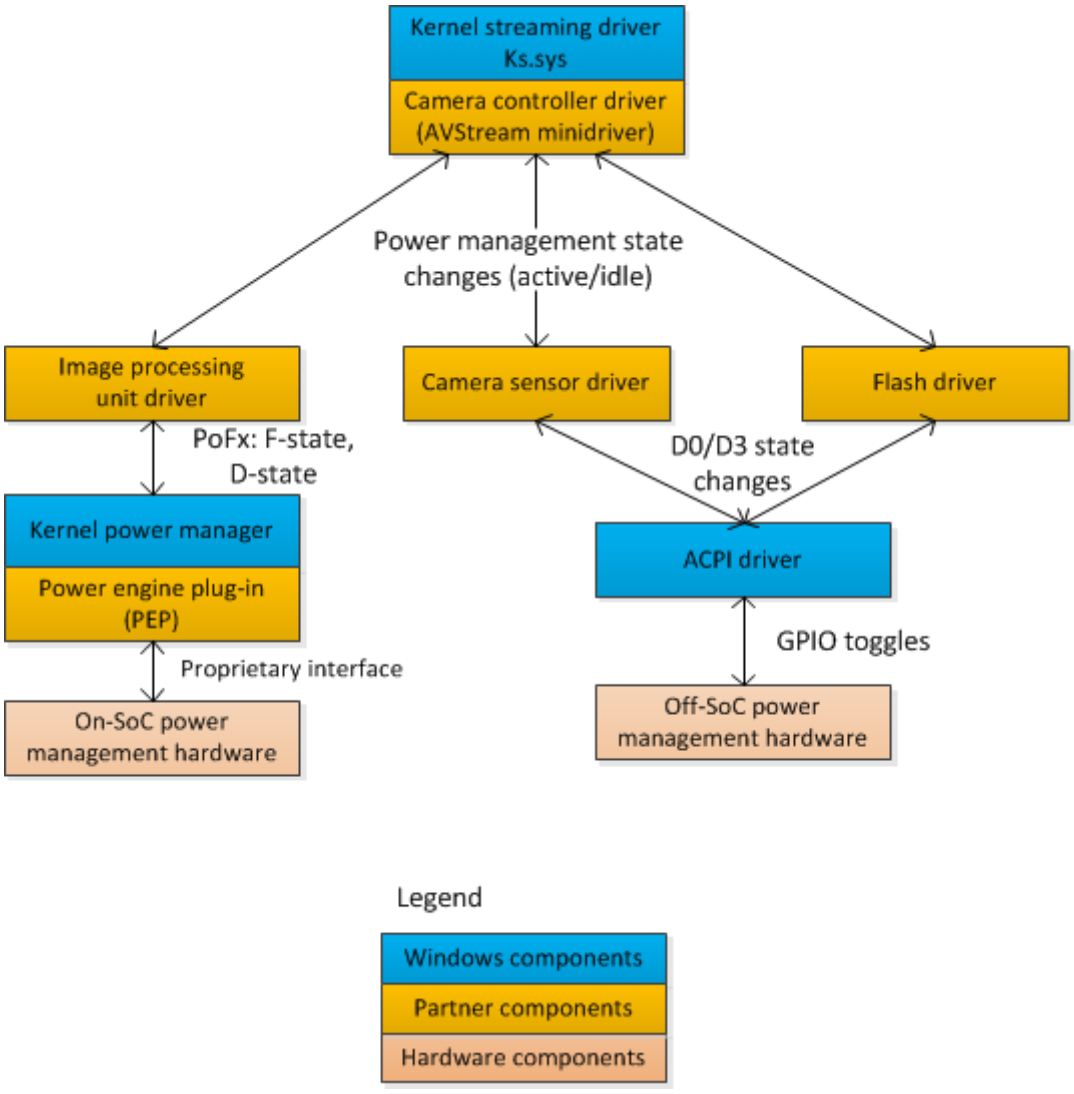
The camera subsystem should be exposed to Windows through a single AVStream minidriver called a *camera controller driver*. We recommend that the camera controller driver not access hardware directly and not directly power-manage hardware components. Instead, the camera controller driver should forward power management and hardware requests to other the drivers that comprise the camera subsystem.

The on-SoC image processing hardware should be power-managed by the SoC power engine plug-in (PEP). The image processing hardware should be managed by a Windows Driver Frameworks (WDF) driver and this driver should enable cooperation with the PEP by setting the **IdleTimeout** member in the **WDF_DEVICE_POWER_POLICY_IDLE_SETTINGS** structure to **SystemManagedIdleTimeout**. This setting enables the PEP to control any clock and

power-rail sharing topology that is unique to the SoC hardware. The driver that controls the image processing unit on the SoC should represent the entire image processing unit as a single power-managed component so that the default WDF capabilities for power management can be used.

The off-SoC camera subsystem components should be managed by one or more Kernel-Mode Driver Framework (KMDF) drivers. The drivers for the off-SoC components must transition to the power-removed (D3) state when their components are no longer required for camera capture. Additionally, the drivers for the off-SoC components must enable D3cold, which allows the underlying ACPI subsystem to change the state of GPIO lines to apply and remove power. For more information, see Supporting D3cold in a Driver.

The following block diagram shows the recommended driver architecture.



All of the drivers that comprise the camera subsystem—including the camera controller driver, the image processing unit driver, and the drivers for the off-SoC camera components—must be enumerated in the same driver installation (.inf) file. All camera subsystem drivers must be members of the imaging PnP device setup class. The ClassGuid for imaging devices is {6bdd1fc6-810f-11d0-bec7-08002be2092f}.

Each driver that represents a single camera component should be enumerated as a single device in the ACPI namespace.

Active and power-removed states

The camera controller driver must transition the camera devices to the power-removed state when no applications are streaming content from the camera device. An application may stop streaming because it has been closed by the user or is transitioned to the background and is suspended.

If an application initiates streaming from a camera whose devices are in the power-removed state, the camera controller driver must transition the camera devices back to the active state within 100 milliseconds.

To change the power states of the various camera subsystem components, camera controller drivers use proprietary interfaces to communicate with the other drivers that comprise the camera subsystem. To query for the appropriate interface, a camera subsystem driver should use the standard method, which is to send an **IRP_MN_QUERY_INTERFACE** I/O request that retrieves a set of function pointers.

The camera controller driver must place the camera device in the power-removed state when all of the streaming pins have entered the **KSSTATE_STOP** state. Windows automatically suspends foreground applications when the user presses the power button and the system enters connected standby. When a capture application is suspended, the camera capture APIs provided by the Windows Runtime are notified and will change the state of the camera capture pins, causing them to enter the **KSSTATE_STOP** state.

When the first streaming pin enters the **KSSTATE_ACQUIRE** state, the camera controller driver must place the camera device—including the on-SoC image processing unit—in the active state.

Associated camera functionality

The camera sensor and flash devices might have additional platform-level functions that must be managed by the driver. These functions can include the following:

- Enabling, disabling, and configuring the camera sensor over the I2C bus.
- Configuring the flash burst rate and brightness level over the I2C bus.
- Detecting thermal conditions from the flash module through GPIO lines from the flash module to the SoC.

To implement these functions, camera device driver developers should use the methods and guidance summarized in the following table.

Function	Description	Hardware/firmware connection	Software mechanism
Sensor configuration	Enumerate the capabilities of the camera sensor hardware or configure its current mode of operation.	Communication over I2C bus. I2C resources are described in the _CRS method under the camera device in the ACPI namespace.	Simple peripheral bus (SPB) input/output (I/O) request interface is used to communicate with I2C host controller and camera sensor device.
Sensor event detection	Raise events or indicate status using GPIO lines from the camera sensor to the SoC.	GPIO resources provided to the camera device. These resources are described in the _CRS method under the camera device in the ACPI namespace. GPIO pins that signal events must be described as GPIO interrupt resources.	Interrupt is processed by driver in response to GPIO event. SPB I/O request interface is used to communicate with sensor device to determine the cause of the interrupt.
Flash configuration	Configure the flash device for burst rate, number of connected LEDs, or other properties.	Communication over I2C bus. I2C resources are described in the _CRS method under the camera device in the ACPI namespace.	SPB I/O request interface is used to communicate with I2C host controller and camera sensor device.
Coordination with image processing unit driver	Initiate and coordinate capture with the image processing circuitry on the SoC.	N/A	Private interface is exposed by the driver that manages the image processing units.

Camera device enumeration

To identify camera devices in the platform, applications typically query the Plug and Play (PnP) manager for instances of camera devices. Each PnP instance corresponds to a single camera device. To identify such an instance, the system integrator defines a camera device in the ACPI namespace. A camera device can stream content to only one

application at a time. However, an application can stream from multiple camera devices simultaneously.

Each camera device that is represented by the camera controller driver (the AVStream minidriver) must be enumerated in the ACPI namespace as a separate device that is a child of the graphics driver.

As a special case, if the SoC platform is not capable of simultaneously streaming content from all camera devices in the platform at any combination of their reported resolutions or modes, a single camera device can be enumerated instead. However, this implementation requires careful consideration and should be undertaken only in direct collaboration with Microsoft.

The devices that represent the remainder of the camera subsystem—including the on-SoC image processing unit and the off-SoC camera sensor, auto-focuser, and flash—should be enumerated as one or more devices in the ACPI namespace. The on-SoC image processing unit should be enumerated as a device that is separate from the devices that represent the off-SoC components of the camera.

Camera power management checklist

System integrators, camera sensor vendors, and System on a Chip (SoC) vendors should use the checklist in this article to ensure that their system power management design is compatible with Windows 10.

- The system integrator must communicate and collaborate with the SoC vendor when selecting camera sensor components and integrating camera devices.
- The camera controller driver developer must do the following:
  - Turn off power to the camera hardware when applications are no longer streaming content from the camera device. (This occurs when all capture pins are in the KSSTATE_STOP state.)
  - Turn on power to the camera hardware when an application starts streaming from any of the capture pins on the camera device.
  - Develop one KMDF driver that manages the on-SoC image processing unit. The camera controller driver should use custom driver interfaces to tell the image processing unit driver to initiate or end camera capture.
  - Ensure the image processing unit driver is registered with the Windows power management framework (PoFx) so that the SoC vendor-provided PEP can control the power management of the image processing unit hardware.
  - Develop one Windows Driver Frameworks (WDF) driver to manage each hardware component that manages the off-SoC camera hardware, including the camera sensor, auto-focuser, and optional flash. The camera controller driver

should use custom driver interfaces to tell the drivers for the off-SoC camera hardware to initiate or end camera capture.
    - Ensure that the drivers that manage the off-SoC camera hardware initiate a D3 transition when the camera components should be powered down so that ACPI is informed of the D3 transition and can remove power from the components. • Ensure that the drivers that manage the off-SoC camera hardware initiate a D0 transition when the camera components should be powered on so that ACPI is informed of the D0 transition and can apply power to the components.
    - Develop any driver code for managing the configuration of the camera sensor hardware or the flash device.
    - Inform the system integrator about the expected ordering of any GPIO and I2C resources that are required to manage the camera sensor hardware or the flash device.
    - Ensure that all of the drivers that comprise the camera subsystem are enumerated in the same driver installation (.inf) file.
    - Ensure that all of the drivers that comprise the camera subsystem are members of the imaging PnP device setup class. The ClassGuid for imaging devices is {6bdd1fc6-810f-11d0-bec7-08002be2092f}.
- The system integrator must design the platform ACPI firmware to do the following:
    - Enumerate each camera device as a separate device in the ACPI namespace.
    - Include a _PLD object under each camera device in the ACPI namespace to indicate whether the camera is on the front or back of the computer.
    - Include a power resource in the root of the ACPI namespace for each camera device. All of the off-SoC hardware components for a given camera device (sensor, AF, flash, and so on) should be in one power resource.
    - Implement the _ON and _OFF control methods for each power resource to change the state of the GPIO pin from the SoC that drives the power rail switching hardware.
    - Provide the _PR0 and _PR3 methods under each camera device in the namespace to reference the power resource for each camera device and its associated hardware.
    - Provide a _CRS object under each camera device in the ACPI namespace to enumerate the GPIO and I2C resources for the camera sensor and flash hardware. The GPIO and I2C resources must be in the order that is specified by the camera sensor driver developer.
- The system integrator should design the platform hardware and power routing so that:
    - Each camera device has its own power rail, which can be controlled independently of the power rails for the other camera devices, and which can be switched on and off by a GPIO pin from the SoC.

- The system integrator must test and verify that:
  - The camera device hardware consumes no power (zero watts) when the camera device is not in use by an application. The system integrator should use instrumented hardware to measure this power consumption.

Fingerprint Sensor

Article • 11/05/2020

WoFPR (Wake on Fingerprint Reader) sensors that support a wake on fingerprint reader capability are wake-capable (and thus maintain a connection to power) during Modern Standby, enabling the ability to initiate captures and therefore wake a PC from Modern Standby.

Expected Behavior

To produce wake on fingerprint reader behavior, follow the below configurations:

Is Modern Standby enabled?	Is the power cut to the sensor?	Resulting WoFPR Behavior
Modern Standby (S0 Low Power Idle)	No	**Enabled**
Modern Standby (S0 Low Power Idle)	Yes	**Disabled**
Traditional Sleep (S3)	Yes / No	**Not Supported**
Hibernate (S4)	Yes / No	**Not Supported**

Additionally, the fingerprint used must be properly enrolled in order to produce wake on fingerprint reader behavior. To enroll a fingerprint, go to Settings > Sign-in options > Windows Hello Fingerprint.

Driver/Firmware/Software Considerations

Do not cut power to the sensor during Modern Standby. To support wake on fingerprint reader, the system must supply power to the sensor during Modern Standby.

Secure sensors will not exhibit WoFPR behavior. Secure sensors are sensors that have implemented the Secure Connection Protocol and have a Microsoft-issued cert to verify that they can operate in a secure mode. This is a subset of the secure biometric feature that includes both face and fingerprint. OEMs decide what hardware to put in their systems and, if they want secure biometrics (which is also a feature of secured-core PCs), they must choose from a list of approved modules. Please refer to Sensor Requirements for Secure Biometrics for more information.

Validation

Validate a touch fingerprint sensor on a Modern Standby system by:

1. Putting the device in Modern Standby (recommended session length is 10+ minutes).
2. Touching fingerprint sensor using a properly enrolled fingerprint.
3. If the system wakes up, WoFPR behavior is working.
4. Check the System Power Report for exit reason 44 (Pdc Signal: Fingerprint) to verify that the correct exit reason was logged.

The system should also pass the Modern Standby Basic HLK test.

GNSS receiver power management for modern standby platforms

Article • 08/24/2021

This topic discusses Global Navigation Satellite System (GNSS) power management for modern standby-capable platforms.

A Windows PC that implements the modern standby power model may also contain a Global Navigation Satellite System (GNSS) device. A GNSS device enables the user to obtain high-precision positioning information from a satellite navigation system, such as the Global Positioning Systems (GPS) or Global Orbiting Navigation Satellite System (GLONASS). After the hardware platform enters modern standby, the GNSS device must enter a low-power mode of operation in which it consumes no more than 1 mW of power. The GNSS device must then stay in this mode until the platform exits modern standby.

GNSS devices that support GPS and the Global Orbiting Navigation Satellite System (GLONASS) have been available for some time, but newer GNSS devices support satellite navigation systems such as BeiDou Navigation Satellite System (BDS) and Galileo.

A modern standby platform is typically built around a System on a Chip (SoC) integrated circuit. Windows supports the following options for adding a GNSS device to such a platform:

- Incorporate a mobile broadband (MBB) module that contains an integrated GNSS device. This method is common in cellular phones.
- Select a SoC that contains an integrated GNSS device.
- Use a low power bus such as I2C, SPI, or UART to connect a stand-alone GNSS device to the SoC.

If possible, the system integrator should select a GNSS device that implements a low-power sleep mode in which the device consumes no more than 1 mW of power. The GNSS device vendor must provide a location sensor driver that translates location data from the GNSS device to the form required by client applications. These applications connect to the GNSS device through the Windows Location API. The location sensor driver for the GNSS device keeps track of the following information:

- The number of clients that are currently connected to the GNSS device through the Location API.
- The state of the radio on/off switch for the GNSS device. The user controls this switch through the Windows Settings application. For more information, see

Integrating radio management.

The location sensor driver uses this information to determine when the GNSS device is idle so that it can be placed in a low-power mode. A GNSS device is actively being used only if one or more clients are connected to the device. Otherwise, the device is idle.

To extend system battery life, access to the GNSS device is restricted during modern standby. Lock-screen applications can access geofencing and position information during modern standby. However, non-lock screen applications can use the Location API to obtain location data from the GNSS device only while the display is turned on and the platform is interacting with the user. Shortly after the display turns off and the platform enters modern standby, any non-lock-screen applications that have connections to the GNSS device are automatically disconnected by Windows, and the applications are suspended.

Power management modes

The GNSS receiver is expected to have 4 power management modes as described below.

GNSS Receiver Power Modes

Device Power Mode	Description	Device Power State (Dx)	Radio State (as shown to user)	Number of Connected Clients	Average Power Consumption (mW)	Transition Mechanism
Active (acquisition)	The GNSS receiver is acquiring a satellite fix.	D0	On	>= 1	~200 mW	N/A
Active (1 Hz Tracking)	The GNSS receiver has acquired a satellite fix and is providing data to one or more applications on a periodic basis.	D0	On	>= 1	~100 mW (device-specific)	N/A

Device Power Mode	Description	Device Power State (Dx)	Radio State (as shown to user)	Number of Connected Clients	Average Power Consumption (mW)	Transition Mechanism
Sleep	The GNSS receiver has no connected clients The GNSS receiver hardware still has power applied to preserve context on the device and reduce resume latency time to the Active mode.	D3(D3hot)	Off or on	0	<1 mW (device-specific)	Software-initiated. This is the selective suspend state for GNSS radios attached directly to the Universal Serial Bus (USB) bus or a USB composite device.
Power removed	The GNSS receiver has no connected clients, the radio is powered down and all power from the GNSS receiver has been removed by an external entity.	D3(D3cold)	Off or on	0	0 mW	Software-initiated and requires hardware coordination to remove power.

The software driver for the GNSS receiver also needs to handle radio management as described below.

Platform implementation requirements

A modern standby platform has multiple options for physically integrating a GNSS receiver. The GNSS receiver may be part of a standalone device which is connected to the SoC by a low-power communications bus. The GNSS receiver may also be integrated into a mobile broadband (MBB) radio as GNSS receivers are common in cellular phones. Finally, the GNSS receiver may be integrated onto the SoC itself.

These multiple options present a challenge to the system integrator when determining which implementation to select for a platform that requires GNSS functionality. For Windows modern standby platforms, system integrators are recommended to integrate GNSS in the following priority order:

1. If the system is equipped with an MBB radio, use the GNSS receiver integrated into the MBB module.
2. If the system is equipped with a SoC that has an onboard GNSS receiver, use the GNSS receiver integrated into the SoC.
3. Integrate a standalone GNSS receiver attached to the SoC over a low power communications bus (such as UART, SPI or I2C).

The system integrator should not expose any extraneous GNSS receivers to Windows. In the presence of multiple GNSS receivers exposed to the operating system, Windows will NOT aggregate the location information from all GNSS devices.

Software power management mechanism

GNSS devices in Windows platforms are expected to be managed by a User-Mode Driver Framework (UMDF) driver that uses the Windows Sensor Class Extension model and implements the ISensorDriver interface. This GNSS driver is called a location sensor driver and is expected to power-manage the underlying GNSS receiver and provide data to application requests for location information.

Overview and application model

Location sensor drivers use the number of connected application clients as the primary mechanism to determine when the GNSS device can enter the sleep or power removed mode. The location sensor driver is also responsible for communicating with a GNSS vendor-provided radio manager library that enables the user to control if the GNSS radio is enabled or disabled. The location sensor driver can use both the number of connected clients and the user's preference for radio state to ensure the GNSS device is in a low-power sleep or power removed mode whenever possible.

Microsoft Store apps manage the lifetime of foreground and background applications and therefore have significant influence on the number of application clients connected to the location sensor driver. A foreground application requests location information through the Windows Location API. When the foreground application is switched to the background by the user, the Microsoft Store app model ensures that the client application is disconnected from the GNSS receiver.

This application model mechanism allows the location sensor driver to easily track the number of connected application clients. Whenever there are zero connected application clients, the location sensor driver can transition the GNSS device into a sleep or power removed mode.

When the system enters modern standby, Windows will automatically suspend all foreground applications resulting in the location sensor driver observing the number of connected clients transition to zero.

Integrating radio management

The location sensor driver must also implement an interface to communicate with the vendor-provided radio manager library. The radio manager library allows Windows to expose a GNSS device radio "on/off" control in the Windows Settings application.

Location sensor driver implementation requirements

The location sensor driver must place the GNSS device into a low-power D3 state when the GNSS device is not in use. The GNSS device is not in use when there are no currently connected clients or the GNSS has been disabled through the radio manager.

The location sensor driver must persist the GNSS device in the D3 state at all times when the GNSS device has been disabled through the radio manager. This requirement prevents the driver from using a power-managed queue and simply forwarding requests from connected clients. The location sensor driver must use a non-power-managed queue for I/O and manage idle directly using the *StopIdle* and *ResumeIdle* methods. The location sensor must continue to be the power policy owner for the driver stack of the GNSS device. The driver must set the value of *bPowerManaged* to FALSE when calling *IWDFDevice::CreateIoQueue* to create the I/O queue.

As mentioned above, the driver uses the number of connected clients and the state of the radio from the radio manager to determine whether the GNSS device is idle. When the GNSS device is idle, the driver calls the ResumeIdle method which causes the driver

framework to initiate a D3 transition. The driver framework will execute the driver's implementation of the *OnD0Exit* method.

When the GNSS device should be reactivated due to a new connected client or power on of the radio, the driver calls the *StopIdle* method. The driver framework will execute the driver's implementation of the *OnD0Entry* method. Note, the driver must execute the *StopIdle* method with the *WaitForD0* parameter set to FALSE.

The state diagram provided below illustrates this relationship and serves as a guide for the driver developer for when to call the *StopIdle* and *ResumeIdle* methods.

Because the driver is responsible for keeping track of whether the device is idle, it must directly manage the I/O queue and resulting hardware device accesses.

The location sensor driver must implement both the *IPnpCallbackSelfManagedIo::OnSelfManagedIoSuspend* and *IPnpCallbackSelfManagedIo::OnSelfManagedIoRestart* methods. Note that the driver framework will call IPnpCallbackSelfManagedIo::OnSelfManagedIoInit when the device is started, including on system boot. Subsequent calls are to the *IPnpCallbackSelfManagedIo::OnSelfManagedIoRestart* callback. These interfaces must be registered when the driver framework calls the CreateDevice method.

The *IPnpCallbackSelfManagedIo::OnSelfManagedIoRestart* signals the location sensor driver that requests into the driver may directly interact with the GNSS device hardware, including requests from *ISensorDriver::* callbacks. Note that the driver framework guarantees that the device hardware may be accessed in both the *OnD0Exit* and *OnD0Entry* methods.

Similarly, when the *IPnpCallbackSelfManagedIo::OnSelfManagedIoSuspend* method is called by the framework, the driver must complete all ISensorDriver requests immediately before returning from *OnSelfManagedIoSuspend*. The driver cannot access the device hardware if doing so might prevent any of these requests from completing within one second. If the location sensor driver has a pending request to the GNSS device hardware, the request must be cancelled if it cannot be completed in any other way without violating this time constraint.

If the driver does not directly interact with a hardware device or all outstanding hardware requests are guaranteed to complete within one second, the driver should implement the *OnSelfManagedIoSuspend* method by using the following procedure:

1. Set a global flag indicating the device is inactive.
2. Call the *StopSynchronously* method on the Windows Driver Frameworks (WDF) queue.

3. Stop any other, asynchronous work that accesses the GNSS device hardware.
4. Call the *Start* method on the WDF queue.
5. Clear the global flag set in step 1.

For a code example that performs these operations, see the OnSelfManagedIoSuspend implementation in Sensors Geolocation Sample Driver (UMDF Version 1).

If the driver interacts directly with a hardware device, then any outstanding requests to the hardware must be cancelled before flushing the I/O queue. The driver should implement the *OnSelfManagedIoSuspend* method by using the following procedure:

1. Set a global flag indicating the device is inactive.
2. Call the *Stop* method on the WDF queue.
3. Cancel all pending hardware requests to allow all *ISensorDriver* callback threads to complete.
4. Call the *StopSynchronously* method on the WDF queue.
5. Stop any other, asynchronous work that accesses the GNSS device hardware.
6. Call the *Start* method on the WDF queue.
7. Clear the global flag set in step 1.

All location sensor drivers must synchronously purge the I/O queue within their implementation of the *OnSelfManagedIoFlush* method.

Sleep and power removed modes

GNSS devices may support both a sleep mode and a power removed mode when the device has local context preserved and can still respond to requests over the communications bus without external signaling. (Typically, a device in power removed mode cannot respond to bus requests.) The location sensor driver must be written to understand if the underlying device is capable of the power removed mode. The location sensor driver should enable D3cold only if the underlying device is capable of the power removed mode and the driver is capable of saving/restoring context and re-initializing the device. Otherwise, the location sensor should continue to use D3 as its idle state, but not enable D3cold. This allows the underlying bus and filter drivers to put the device in the sleep low-power mode and not the power removed mode.

When the location sensor driver indicates that it supports D3cold and begins a D3 transition, the underlying bus and filter drivers are responsible for removing power from the device. The mechanism may be implemented by ACPI in the case of UART-connected GNSS devices. Or, the mechanism may be enabled by a combination of the USB hub driver and the ACPI driver in the case of USB-enumerated GNSS devices.

If the GNSS device is on the SoC, proprietary driver and firmware from the SoC vendor implemented in the underlying driver(s) is responsible for removing power from the GNSS device. If the GNSS device consumes greater than 1 mW in its sleep power mode, the GNSS driver, platform firmware and hardware must be engineered to place the device in the power removed mode when all clients are disconnected.

Idle detection

The location sensor driver for a GNSS device must transition the device to the sleep power mode whenever possible. If an app requests a long report interval, the location sensor driver should transition the GNSS device to the sleep power mode until the next fix is requested. The location sensor driver must transition the GNSS device to the active power mode with sufficient time to triangulate a fix and provide the app with location data.

For example, assume the shortest requested report interval is 30 minutes and the device requires one minute to warm up and acquire a location fix. In this scenario, the location sensor driver must:

- Immediately after providing location information, call ResumeIdle which will transition the GNSS device to sleep mode (D3).
- Arm a timer to expire 28 minutes in the future. (TimerExpiration = ReportInterval – WarmUpTime).
- When the timer expires, call StopIdle which will transition the GNSS device to D0.
- When the device has acquired a fix, provide location information to the application. **Note** The location sensor driver must carefully configure D3cold.

If the device requires continual power in order to achieve the resume latency for the WarmUpTime, D3cold must not be enabled. D3cold can be enabled dynamically at runtime by changing the value of ExcludeD3Cold in the WDF_DEVICE_POWER_POLICY_IDLE_SETTINGS structure.

When a USB-attached GNSS device transitions to the sleep mode (D3) with D3cold disabled, the device will transition to the USB suspend state which saves significant chipset and processor power. If the location sensor driver enables D3cold and transitions to the sleep mode (D3), the underlying platform may remove power from the device even when it is attached via the USB bus.

Supported hardware configurations

Windows supports four physical hardware configurations for the GNSS device. The connectivity bus differentiates each hardware configuration.

GNSS connected to the SoC over UART

In this configuration, the GNSS radio is a standalone device connected to a UART on the SoC. The GNSS radio may have one or more GPIOs between the radio and the SoC for the purposes of transitioning between the active and sleep power modes or for handling reset and power-up sequencing conditions.

If the GNSS radio consumes less than 1 mW in its sleep power mode, the GNSS radio may be connected to any system power rail that meets the device's specifications.

The GNSS device must be declared in the ACPI namespace and GPIOs for power management should be controlled by the _PS3 and _PS0 control methods under the device in the ACPI namespace. The _PS3 and _PS0 methods are executed by the ACPI driver in response to the D3 and D0 transitions initiated by the location sensor driver. The system integrator must declare the GPIOs as part of a GPIO operation region in the ACPI namespace.

If the GNSS receiver consumes more than 1 mW in its sleep power mode, the GNSS receiver must be connected to a power rail which can be switched on and off using a GPIO controlled by ACPI firmware on the SoC. In this configuration, the location sensor driver must enable D3cold. The GPIO for controlling the switchable power rail must be exposed in an ACPI GPIO operation region. The system integrator must describe a power resource for the switchable power rail, including the _OFF and _ON methods, as well as the references to the power resource under the GNSS device in the namespace.

The Windows ACPI driver will evaluate the _OFF method when the location sensor driver transitions to D3. When the location sensor driver transitions to D0, the Windows ACPI driver will evaluate the _ON method under the power resource. The implementation of the _OFF and _ON methods should toggle the GPIO that controls the switchable power rail and implement any power sequencing delay(s) required.

GNSS integrated into the SoC

If the GNSS device is physically integrated into the SoC, the implementation of communications and power management is specific to the SoC itself.

The GNSS device must still be enumerated via ACPI, though communications to the underlying GNSS receiver may occur through a transport driver that is provided by the SoC vendor. In this configuration, the location sensor driver must still implement a D3

transition to enter the sleep or power removed mode when all clients have disconnected. The D3 transition ensures that Windows OS power management tools and diagnostics can easily observe the power state of the GNSS device.

System integrators that plan to leverage a GNSS radio integrated into the system SoC should consult closely with the SoC vendor for firmware, driver and power management support.

GNSS integrated into a USB-attached MBB radio as a USB composite device

The system designer may choose to integrate a USB-attached MBB module that contains an embedded GNSS receiver. In this hardware configuration, the location sensor driver communicates with the embedded GNSS receiver directly over the USB bus as one function in a USB composite device.

Note Systems with GNSS devices on the MBB module require careful integration consideration. Contact your Microsoft representative to review the hardware, software and firmware design for these solutions.

In this configuration, the GNSS device is an integrated part of the MBB module. The GNSS radio may share processing, power supply and RF antenna components with the MBB radio. The GNSS radio is exposed directly to software as one interface on a USB composite device. The location sensor driver communicates with the GNSS radio directly over the USB bus using USB driver interfaces implemented inside the location sensor driver.

Power management of the GNSS hardware is driven by in-band communications to the GNSS module. The GNSS module and the location sensor driver must support an in-band mechanism to turn on and off the GNSS radio. This mechanism must not rely on the use of any GPIOs from the SoC to the MBB+GNSS module.

Similarly, the GNSS module and the location sensor driver must support transitioning the device to the D3 state such that the USB composite device can enter the USB suspend (selective suspend) state. All functions on a USB composite device must be suspended in order for the composite device to suspend. The GNSS device must be in the Sleep (D3) mode in order for the GNSS function and the whole USB composite device to be in the suspend state.

Note The GNSS device and the driver must support selective suspend otherwise the USB host controller on the SoC cannot enter a low-power mode and will prevent power savings during modern standby.

In this configuration, the GNSS device is enumerated via USB and the USB composite driver, but is described in the ACPI namespace. In this configuration there is no support for GPIO communications between the GNSS device on the MBB module and the SoC. The GNSS device must remain enumerated to Windows via ACPI during the entire time that the platform stays in the S0 system power state, even if the radio is disabled by the user. The GNSS device must not ever disappear from the USB bus during any portion of system on time.

GNSS integrated into a USB-attached MBB radio using device services

The system designer may choose to integrate a USB-attached MBB module that contains an embedded GNSS receiver. In this hardware configuration, the location sensor driver communicates with the embedded GNSS receiver through the mobile broadband device services interface as opposed to the GNSS device being exposed as a standalone USB function as part of the composite device that represents the whole MBB module.

Note This configuration is not recommended. System integrators choosing this method of GNSS device integration should contact their Microsoft representative to validate correct implementation. Exposing the GNSS device as part of a USB composite device representing the MBB module is preferred.

In this configuration, the GNSS device is an integrated part of the MBB module. The GNSS radio may share processing, power supply and RF antenna components with the MBB radio. The GNSS radio is exposed indirectly to software via the device services interface which can be accessed using the WindowsIMbnDeviceServices COM interface. The location sensor driver communicates with the GNSS radio through the IMbnDeviceServices interface.

Power management of the GNSS hardware is driven by in-band communications over the device services interface to the MBB module. The GNSS hardware must support an in-band mechanism over the device services interface to turn off the radio and place the GNSS device in a low-power mode. These mechanisms must be accessible by the location sensor driver through the device services interface.

In this configuration, the GNSS device must be enumerated by ACPI and described in the ACPI namespace as a child of the mobile broadband module. The GNSS device will have no hardware resources described under the device in the ACPI namespace.

The location sensor driver must still perform the same set of power management implementation guidance as described in the previous driver implementation

requirements section.

In this configuration there is no support for GPIO communications between the GNSS device on the MBB module and the SoC. All power management and radio communications are performed physically over the USB bus and exposed to the location sensor driver through the device services interface. The GNSS device must remain enumerated to Windows via ACPI for all of system on time, even if the radio is disabled by the user.

When implementing this hardware configuration, the system integrator is encouraged to partner closely with the MBB module vendor to ensure the GNSS device is exposed correctly in the ACPI namespace.

Wake concerns

There are no wake concerns for GNSS devices. GNSS devices are not expected to support waking the SoC from a low-power state.

Testing and validation

GNSS device vendors, MBB module vendors and system integrators should follow these recommendations to test and validate the power management implementation of the GNSS device and its related software components. For more information, see Global Navigation Satellite System (GNSS) Test Guidance.

Location sensor power management

The system integrator should validate that the location sensor driver for the GNSS device performs power management transitions and enters the D3 state when all clients have disconnected or the radio has been disabled.

The basic test cases are:

- Observe the location sensor driver transitions to D3 within 10 seconds of the screen turning off for modern standby.
- Observe the location sensor driver transitions to D3 within 10 seconds of the radio being turned off in the Windows Settings application.
- Observe the location sensor driver transition to D0 immediately after exiting modern standby and launching an application that uses the Location API.

The easiest way to observe Dx state transitions of the location sensor driver is to use the Windows Performance Toolkit to compare against to other Windows sensor devices. This method uses Windows instrumentation to validate the D3 IRP is transitioning through the device driver stack that comprises the GNSS device. The Windows power manager includes built in Event Tracing for Windows (ETW) instrumentation, including instrumentation for device power (Dx) IRPs. To view this information in a manual mode, obtain and install the Windows Performance Toolkit (WPT) on the system under test.

Begin a user-mode XPerf trace by using the following commands:

1. Open an Administrator Command Prompt.
2. Browse to the **\%ProgramFiles%\Windows Kits\8.0\Windows Performance Toolkit** folder.
3. Start Xperf: `xperf.exe -start power_session -on Microsoft-Windows-Kernel-Power`
4. Transition the system to modern standby by using the power button.
5. Wait 120 seconds.
6. Transition the system out of modern standby using the power button.
7. Wait 60 seconds.
8. Run the following command to stop the event logging: `xperf.exe -stop power_session`
9. Convert the binary trace file into .csv and human-readable format: `xperf.exe –i \user.etl > power.txt`
10. Open the power.txt file in a text editor and search for the hardware ID of the GNSS device. The hardware ID of the GNSS device can be determined from the **Details** tab of the device properties in **Device Manager** under *Device Instance Path*. In the example below, the device instance path of the GNSS device is *ACPI\MST0731\2&daba3ff&0*.
11. The D3 IRP for the GNSS device is indicated by an event of type Microsoft-Windows-Kernel-Power/IRP/Stop with the device instance path of the GNSS device and the last event value of 3 for the D3 state. The event output below from the power.txt file shows the start of the D3 IRP.

    ```
    Microsoft-Windows-Kernel-Power/Irp/Start , 7605393, "Unknown" ( 4), 256, 0, , , , , 0x868e2728, 1, 2, 0x85fb56e0, 25, "ACPI\MSFT0731\2&daba3ff&0", 3
    ```

12. This event should be logged near the beginning of the power.txt output file. The parameter *0x868e2728* in the example above is a pointer to the D3 IRP structure. By finding the subsequent events with this same IRP pointer, a view of the D3 IRP flowing through the driver stack that comprises the GNSS device can be discovered. Note that the IRP pointer will be system and boot-lifetime specific.

13. `Microsoft-Windows-Kernel-Power/Irp/Start , 7605393, "Unknown" ( 4), 256, 0, , , , , 0x868e2728, 1, 2, 0x85fb56e0, 25, "ACPI\ATML1000\2&daba3ff&0", 3`

14. `Microsoft-Windows-Kernel-Power/Driver/Start , 7605416, "Unknown" ( 4), 20, 0, , , , , 0x868e2728, 0x85fb56e0, "\Driver\gpsdrv"`

15. `Microsoft-Windows-Kernel-Power/Driver/Stop , 7605515, "Unknown" ( 4), 20, 0, , , , , 0x868e2728, 0x85fb56e0`

16. `Microsoft-Windows-Kernel-Power/Driver/Start , 7608351, "Unknown" ( 4), 20, 0, , , , , 0x868e2728, 0x857ffb90, "\Driver\ACPI"`

17. `Microsoft-Windows-Kernel-Power/Driver/Stop , 7608416, "Unknown" ( 4), 20, 0, , , , , 0x868e2728, 0x857ffb90`

18. `Microsoft-Windows-Kernel-Power/Driver/Start , 7608424, "Unknown" ( 4), 20, 0, , , , , 0x868e2728, 0x85fb56e0, "\Driver\sensdrv"`

Validating the GNSS device is returning to D0 when the screen is turned on is a similar process. A Microsoft-Windows-Kernel-Power/IRP/Start event for the GNSS device will be logged with a 0 (D0) target state. The D0 IRP will flow through the driver(s) comprising the GNSS device stack in a similar manner to the D3 IRP.

GNSS power management checklist

System integrators, GNSS radio vendors, and SoC vendors should use the following checklist to ensure that their system power management design is compatible with Windows 8 and above.

- Integrate a GNSS device into the modern standby-capable platform in the following configuration preference order:

    1. Integrated together with a MBB module (for systems equipped with MBB), connected over USB.
    2. Integrated onto the SoC (for systems that have GNSS on the SoC).
    3. Standalone outside the SoC connected to UART, I2C or another low-power bus.

- Select a GNSS device that has a sleep (radio off) average power consumption of less than 1 mW, including any bus connection interfaces.
- If the GNSS device has a sleep (radio off) average power consumption of greater than 1 mW, the system integrator and GNSS device vendor must support removing power completely from the GNSS device when there are no connected application clients or the radio is turned off by the user.
- Ensure that the GNSS device vendor provides a location sensor driver that implements runtime power management based on the number of connected clients and the state of the GNSS radio.
- Ensure that the GNSS device vendor provides a radio manager library that exposes the GNSS radio in the Windows Settings application.

  The location sensor driver must implement a private interface to communicate radio on/off state between the vendor-provided radio manager library and the vendor-provided location sensor driver.
- If the GNSS is a standalone device outside of the SoC connected via UART, I2C or other low-power bus, the system integrator and GNSS device vendor must:

  1. Document any GPIOs between the GNSS device and the SoC itself.
  2. Expose any GPIOs for power management as part of a GPIO operation region in the ACPI namespace.
- If the GNSS is a standalone device outside of the SoC connected via UART, I2C or other low-power bus and the average power consumption of the GNSS device in the sleep power mode is greater than 1 mW, the system integrator and GNSS device vendor must:

  1. Provide an ACPI power resource and _ON/_OFF control methods to describe the GNSS power domain.
  2. Provide the _PR0 and _PR3 methods under the GNSS device in the ACPI namespace that refer to the described ACPI power resource.
  3. Ensure that the location sensor driver does a D3 transition and enables D3cold in the driver.
- If the GNSS is part of a MBB module, the system integrator and GNSS device vendor must:

  1. Expose the GNSS device as part of a USB composite device.
  2. Provide a location sensor driver that communicates to the GNSS device directly through the USB bus.

3. Ensure that radio on/off and all power management of the GNSS device may be performed in-band over the USB bus. No GPIOs can be used to change GNSS radio or power state in this hardware configuration.
4. Ensure that the USB device describing the MBB module or the USB composite device describing both the MBB and GNSS radio enters the USB suspend state during modern standby.
5. The location sensor driver MUST enter the D3 (sleep) mode when all clients are disconnected or the radio has been turned off even if communicating to the device over the device services interface.
6. If the GNSS device is controlled over the mobile broadband device services interface (which is not recommended), the GNSS device must be described in the system ACPI namespace with no hardware resources. The GNSS device must be described as a child of the mobile broadband module in the ACPI namespace.

- Test and validate that the GNSS device and the location sensor driver perform power management correctly. Please validate the following test cases at a minimum:
  - Observe the location sensor driver transitions to D3 within 10 seconds of the screen turning off for modern standby.
  - Observe the location sensor driver transitions to D3 within 10 seconds of the radio being turned off in the Windows Settings application.
  - Observe the location sensor driver transition to D0 immediately after exiting modern standby and launching an application that uses the Location API.

- Validate the power consumption of the GNSS device in the sleep (D3) state and ensure that it is less than 1 mW on average.

Supporting dGPUs on Modern Standby Systems

Article • 05/19/2021

Overview

Discrete GPUs present a challenging problem for Modern Standby designs, as they put strain on the platform both in terms of power consumption and resume from sleep latency. It's a fundamental power vs. performance tradeoff—if the dGPU keeps its memory in VRAM (either by staying active in D0 or by enabling VRAM self-refresh capabilities in D3), then it consumes additional power—a problem in the face of upcoming power regulations. However, if the dGPU enters a D3 state and fully offloads its contents to power down VRAM, the system could suffer from extended latencies when resuming from sleep, as it can take several seconds to restore VRAM contents from main memory. This tradeoff must be balanced to ease adoption of Modern Standby for systems with dGPUs and provide the best user experience possible. This document aims to explain the dGPU problem in detail and outline guidance for supporting dGPUs in Modern Standby systems.

Current dGPU Support

The problem of supporting dGPUs can be thought of along two separate categories—(1) add-in dGPU cards and (2) soldered-in dGPUs. The following chart outlines the situation and guidance across these scenarios. The remaining sections expand upon the information and requirements highlighted in this table.

As the ecosystem works towards better dGPU power & performance: :

Plug-in dGPUs: dGPUs that are plugged into an open PCIe slot, either in a shipping configuration or added independently by an end user. This applies to any system with an open PCIe slot capable of supporting an add-in dGPU card.	• Systems must implement the PCI ECN to allow the dGPU to enter its D3cold state • The dGPU must be capable of entering D3cold • Self-refresh support in the dGPU is optional—if supported, the system provides a better user experience by providing the "Instant On" capability, power regulations should be considered. • Threshold for the OS using the dGPU's self-refresh capability is 300 MB in VRAM

	• If the dGPU is not using self-refresh (lacks support or feature disabled in the graphics driver), Microsoft's Directed PoFx (DFx) framework will force the dGPU down from D0 after 2 minutes in idle—VRAM will be offloaded • Systems with dGPUs are exempted from the requirement in HLK testing for a 1 second resume latency from Modern Standby since offloading VRAM causes a latency hit
Soldered-down designs: This covers both high end notebooks or all-in-ones shipping with a dGPU and hybrid systems that use both an integrated and discrete GPU.	• The dGPU must be capable of entering D3cold. • Self-refresh support in the dGPU is recommended—if supported, the system provides a better user experience by delivering the "Instant On" capability, but the system must be carefully engineered to meet power regulations. • Threshold for the OS using the dGPU's self-refresh capability is 300 MB in VRAM • If the dGPU is not using self-refresh (lacks support or feature disabled in the kernel mode graphics driver), Microsoft's Directed PoFx (DFx) framework will force the dGPU down from D0 after 2 minutes in idle—VRAM will be offloaded • Systems with dGPUs are exempted from the requirement in HLK testing for a 1 second resume latency from Modern Standby since offloading VRAM causes a latency hit.

Essentially, there are a couple of main points regarding dGPU support in Modern Standby designs:

- System manufacturers should first ensure their systems meet power regulations, and then optimize for the best possible user experience with respect to resume time from modern standby.
- The ecosystem is moving towards self-refresh VRAM being a requirement for maintaining a great user experience with lower dGPU power consumption.
  - It is in the ecosystem's best interest to invest in improving GDDR power consumption.

dGPU VRAM Self-Refresh Behavior

This section discusses the current heuristics around dGPU self-refresh behavior—system designers should take this into account when evaluating their system behavior and performance, as these will be dependent on the scenario—specifically, it will depend on the dGPU's self-refresh capability and how much content is currently held in VRAM.

Beginning with Windows 10, the operating system is smart about making the decision for when to use self-refresh and when to not use self-refresh. If the VRAM is relatively empty when entering sleep, it will be powered off without making use of self-refresh. Otherwise, self-refresh VRAM will be used. The threshold for this behavior is currently defined as 300 MB of contents in VRAM and may be further optimized in the future. The following table describes the current self-refresh heuristics:

	Entering Modern Standby with <= 300 MB of VRAM in use	**Entering Modern Standby with > 300 MB of VRAM in use**
dGPU with self-refresh VRAM support	• VRAM is evicted • D3cold with VRAM off	• VRAM is preserved • D3cold with VRAM in self-refresh
dGPU without self-refresh VRAM	• VRAM is evicted • D3cold with VRAM off	• VRAM is preserved • dGPU stays in D0 • Microsoft's Directed PoFx (DFx) framework will force the dGPU into D3cold and evict VRAM after 2 minutes

Add-in dGPU card solution

This section outlines the requirements of supporting an add-in dGPU card in a Modern Standby desktop system design on a high level. For implementation specifics, refer to hardware vendor guidance.

When building a Modern Standby desktop system with support for an add-in dGPU card, there are a few key requirements that must be included for a complete solution. These requirements span across dGPU design, dGPU drivers, motherboard design, and firmware implementation.

Requirement	**Description**	**Resources**

Requirement	Description	Resources
The BIOS implements the _DSM specified in the attached PCI ECN	These functions allow for the PCIe device driver to negotiate with the platform for the aux power necessary for the dGPU card to support self-refresh in D3cold.	_DSM Additions for Runtime Device Power Management
The SoC can provide up to 1A of auxiliary power to the PCIe slot	This is specified in the ECN to the PCI spec. dGPU power consumption varies by card, so the SoC must be capable of providing the necessary power to support self-refresh on cards, which varies up to 1A.	Refer to hardware vendor's implementation guidance
The dGPU card supports self-refresh VRAM and D3cold	Self-refresh VRAM allows the dGPU card to enter D3Cold while still maintaining its memory contents for a short resume latency. This is essential to the Modern Standby promise of an “Instant On” experience.	Refer to dGPU designer’s implementation guidance

Networking power management for Modern Standby platforms

Article • 03/17/2021

To support modern standby, Windows is designed to use the right network adapter at the right time for an Internet connection. All modern standby PCs have a Wi-Fi adapter, but some PCs also have a mobile broadband (MBB) adapter and/or a wired Ethernet adapter. During modern standby, Windows automatically connects to the best available network.

Automatic network selection

Windows automatically connects to the best available network during modern standby to help ensure that the system always has a fast, inexpensive, and reliable connection to the Internet.

Windows assumes that the network devices that are integrated into a modern standby system are capable of protocol offloads and wake-on-LAN (WoL) patterns. If a network device does not have both of these capabilities, the network device is disabled during modern standby.

By default, Windows prefers to use a wired Ethernet connection at all times if available, including when the system is in modern standby. If the Ethernet adapter supports protocol offloads and WoL patterns and is connected to a hub or switch, Windows prefers to use this adapter for the Internet connection over all other types of available network devices. If the Ethernet adapter does not support protocol offloads and WoL patterns, the Ethernet adapter is disabled during modern standby and Wi-Fi is used instead.

As long as a Wi-Fi connection to an associated access point is available, Windows uses this Wi-Fi connection and powers down an MBB device if one is present in the system. As soon as the Wi-Fi connection is lost, Windows powers on the MBB device and tries to connect to the cellular network.

After the Wi-Fi connection is lost, Windows tells the Wi-Fi device to keep searching for previously used access points by using the network list offload (NLO) feature. If a previously used Wi-Fi network is discovered, MBB is disconnected and powered down and Wi-Fi is used again.

Network quiet mode

Windows allows periodic execution of third-party system services during modern standby to maintain compatibility with existing desktop applications.

However, the network quiet mode (NQM) feature does not allow these services to access the network during modern standby. NQM simulates a disconnected network for these third-party services, to which the Wi-Fi device appears to be simply disconnected from the Internet.

Desktop and system service application developers must be aware of NQM and not make their applications or services dependent on network access during modern standby.

Similarly, system designers and application developers must be aware that the network devices in modern standby are predominately operating with WoL patterns and protocol offloads enabled. This means that remote access into the system over the network is not available during modern standby, with the exception of Microsoft Store apps that have been pinned by the user to the lock screen.

Network list offload (NLO) for Wi-Fi

The Wi-Fi and MBB devices in a modern standby platform are expected to be highly autonomous. A key Wi-Fi device feature for modern standby is called network list offload (NLO). During modern standby, a Wi-Fi device that supports NLO can automatically connect to previously used Wi-Fi access points.

NLO allows the Wi-Fi device to roam between previously used access points while the SoC remains in the low-power idle mode. As the user commutes between home and work, Windows automatically connects to Wi-Fi, thereby allowing the system to be already connected when the user presses the power button. The user no longer has to wait to connect to Wi-Fi. Instead, Wi-Fi is connected before the user turns on the system. Similarly, email is already downloaded, and connections to Skype and other communications services are automatically reconnected.

For more information about NLO, see Wi-Fi Network List Offload.

Expected network behavior

When a system enters disconnected standby, no patterns are configured for wake across LAN, WiFi, or Mobile broadband. A media disconnect is sent to WiFi and Mobile Broadband parts.

In this section

Mobile broadband (MBB) power management for modern standby platforms	A mobile broadband (MBB) device provides a mobile computer with a wireless data connection to a cellular service. An MBB device supports one or more cellular radio technologies, such as GSM, 3G, CDMA, or LTE. MBB devices in modern standby platforms are all required to provide the same set of power management capabilities and to implement the same general hardware configuration, regardless of the specific cellular technologies supported.
Wi-Fi power management for modern standby platforms	In a Windows hardware platform that implements the modern standby power model, the Wi-Fi device maintains its connection to the network while the platform is in the modern standby power state. Correct power management of the Wi-Fi device—both during modern standby and when the display is turned on—is essential to achieving long battery life.

Mobile broadband (MBB) power management for modern standby platforms

Article • 08/24/2021

A mobile broadband (MBB) device provides a mobile computer with a wireless data connection to a cellular service. An MBB device supports one or more cellular radio technologies, such as GSM, 3G, CDMA, or LTE. MBB devices in modern standby platforms are all required to provide the same set of power management capabilities and to implement the same general hardware configuration, regardless of the specific cellular technologies supported.

The MBB device is expected to use Universal Serial Bus (USB) to connect to the modern standby platform, and to use software provided by Microsoft for all connectivity and power management operations. Windows includes an inbox Mobile Broadband Class Driver, which provides a standardized interface for mobile broadband data transfer, connection management, and power management for MBB radios. The following guidance focuses on power management for MBB devices that are integrated inside the mobile computer chassis. MBB devices that connect to an external USB port are not discussed.

During modern standby (when the screen is off), the MBB device is expected to be in a low-power state. How much power the MBB device consumes in this low-power state depends on whether the user has provisioned the MBB device on the cellular network. If the MBB device is provisioned and the radio in the device is currently enabled by the user for a data connection, the device should be in a low-power connected-sleep mode in which the device's USB function is in the D2 (suspend) device power state. However, if the MBB device has not been provisioned on the network or the user has disabled the radio for a data connection, the device should be in a low-power radio-off mode. In radio-off mode, the MBB device has only enough power applied to respond to host commands over the USB interface.

The implementation of MBB device power management for a modern standby platform is based on the following:

- The user-controlled radio power state of the MBB device.
- The USB bus suspend and resume transitions.

The MBB device must be able to enter a low-power D2 (suspend) state after the MBB radio is turned off and the USB bus interface enters the suspend state. All sleep and

wake power transitions must be signaled over the USB bus. There is no support for out-of-band GPIO signaling to initiate MBB device power transitions or to interrupt the main processor on the System on a Chip (SoC) or core silicon.

If the radio in the MBB device is currently enabled by the user for a data connection, the device must be able to use in-band, USB resume signaling to wake the SoC or core silicon from modern standby. The SoC or core silicon must be able to wake from its lowest power state in response to in-band, USB resume signaling from the MBB device.

Power management modes

The MBB device is expected to support five power-management modes. These modes are a combination of provisioned, connectivity, and radio power states. A transition from one mode to another is communicated to the device directly over the USB bus through commands from the Mobile Broadband Class Driver or USB device state transitions. Transitions between power management modes must not use external GPIO signaling.

The five power-management modes are:

Active

The radio is actively transmitting data or is actively connected to the cellular network.

Connected-sleep

The radio is provisioned on the network and a user account is enabled. The platform is in modern standby. The MBB device is waiting for data from the network to wake up the SoC, and also for events from the SoC. Average across 2G, 3G, LTE, and various DRX modes.

Radio-off

The radio is provisioned on the network, but Windows or the user has turned off the radio in the MBB device.

No-subscription

The user does not have an active subscription.

No-SIM

The device has no SIM.

The following table compares the five power-management modes.

Power management mode	Radio power state	USB device power state (Dx)	Average power consumption	Exit latency to active
Active	On	D0	Scenario-specific	N/A
Connected-sleep	On	D2 (selective suspend)	<= 15 milliwatts (device-specific)	USB D2 -> D0 <= 400 milliseconds (USB specification)
Radio-off	Off	D2 (selective suspend)	<= 5 milliwatts	USB D2 -> D0 < 400 milliseconds + Base station acquisition and registration (specific to location and carrier)
No-subscription	Off	D2 or D3	<= 3 milliwatts	N/A
No-SIM	Off	D2 or D3	<= 3 milliwatts	N/A

Note In no-subscription mode and no-SIM mode, an MBB device can be in either the D2 or D3 device power state, depending on whether the device supports D3.

The power consumption of the MBB device in each of the power modes shown in the preceding table will vary by radio implementation, network type, and distance from the cellular access point. Contact the radio manufacturer for information about the device-specific power consumption in each power management mode shown in the preceding table.

Software power management mechanisms

The two primary software power management mechanisms for the MBB device are the radio power state and the USB function power state.

Radio power state

The Windows Mobile Broadband Class driver will send a MBIM_CID_RADIO_STATE message to the MBB device with a command to turn the radio power state to on or off per the user's preference. When the user disables the radio, the Mobile Broadband Class driver will send the MBIM_CID_RADIO_STATE message with an MBIMRadioOff command.

USB selective suspend

The Windows Mobile Broadband Class Driver is a Windows Network Driver Interface Specification (NDIS) miniport driver. The Mobile Broadband Class Driver and NDIS collaborate to manage the power of the MBB device. Windows will transition the USB function of the MBB device to the USB suspend (or, in Windows terminology, selective suspend) state when the MBB device should be in a low-power mode. The transition to the USB suspend state is independent of the radio power state.

Windows will transition the MBB device to *connected-sleep mode* (USB suspend (D2) state) when the system enters modern standby and the device is provisioned on the network. During connected-sleep mode, the radio is on, and the MBB device is expected to generate USB resume wake signaling and return to the D0 state when the radio receives new data from the network that matches programmed wake patterns or when other enabled wake events occur. The power consumption in connected-sleep mode will vary by cellular technology and distance from the cellular access point.

Windows will transition the radio to *radio-off mode* (USB suspend (D2) state) when the system enters modern standby and the device is provisioned on the network, but the user has set the radio power state to *off*. In radio-off mode, the MBB device is expected to consume no more than five milliwatts on average. The following block diagram shows the control path for initiating the MBB device's transition to the USB suspend state.

If the MBB device has not been provisioned on the network by the user, Windows will transition the device to the *no-subscription mode* (D2 or D3 state) when the platform enters modern standby. When the device is in no-subscription mode, its radio power state is always off. When the MBB device is in the no-subscription mode during modern standby, the power consumption of the MBB device is expected to be equivalent to that of the radio-off mode. However, the MBB device must still remain physically enabled on the USB bus during the entire time that the device is in the no-subscription mode.

At all times—regardless of whether the platform is in modern standby—power must not be removed from the USB function portion of the MBB device. Otherwise, when power is removed, the device will fall off of the USB bus and cause the device to be reported as surprise-removed.

Note On modern standby platforms, there is no support for the execution of third-party driver or ACPI firmware to manage transitions of the MBB device between the power management modes previously described. All power management of the device must be done in-band through the USB bus.

Third-party driver and ACPI firmware is supported for Specific Absorption Rate (SAR) sensors, which change the MBB radio transmitter power in reaction to the proximity of the MBB antenna to the user.

Supported hardware power configurations

To meet the power management requirements of a modern standby platform, only one hardware configuration is supported—the MBB device must use USB to connect to the platform. Additionally, the USB-connected MBB device must:

- Report that it is self-powered and remote-wake-capable in the USB configuration descriptor for the device.
- Be attached to a system power rail that will be powered on at all times regardless of whether the platform is in modern standby.

Because there is no support for the execution of third-party driver or ACPI firmware for MBB device power management, there must be no GPIO lines connected from the MBB device to the SoC for use by the operating system or third-party driver software. All power management of the MBB device must be communicated in-band over the USB bus.

MBB module reset

The MBB device must undergo a reset when the platform enters the ACPI S5 (shutdown) system power state. This requirement exists so that the user can indirectly reset the MBB device if it is not responsive by selecting Restart from the Windows Power menu. In addition, this requirement ensures that the MBB device can be reset to detect a newly inserted or removed SIM.

The system integrator should ensure that the system resets the MBB device using one of the following methods:

- **Reset in UEFI firmware**: The system UEFI firmware can reset the MBB device on system boot. Note that Windows requires platforms that have non-rotational (solid-state) storage to complete power-on self test (POST) within two seconds. Resetting the MBB device must not cause this time to be exceeded.
- **Remove power with switchable power rail**: The system integrator can design the platform power routing to remove power from the MBB device when the platform enters the S5 state. The removal of power from the device be accomplished by using a GPIO-controlled power switch or by placing the MBB device on a power rail that is disabled by the system power-management IC (PMIC) when the platform enters the S5 state.

Wake concerns

An MBB device that is in the USB suspend state must be able to wake the SoC by using USB resume signaling. The SoC must be able to wake from its lowest run-time idle state on detection of a USB specification-compliant resume-from-suspend signal from an attached USB device.

A USB-connected MBB device that is located inside the computer chassis will typically be self-powered. That is, the device draws its power from a system power rail instead of from the USB host controller to which it is attached. The reason for this requirement is that some MBB devices (depending on the cellular technology) require more than the USB specification-allotted suspend current to stay connected to the network during connected-sleep mode.

Testing and validation

The MBB device vendor, the system integrator, and the SoC designer should test the MBB device's USB power management and verify that it is functioning correctly. Specifically, the MBB device should transition to the D2 (USB suspend) device power state and stay in this state during modern standby when the device is not actively transmitting or receiving data. In addition, verify that the MBB device can use in-band USB resume signaling to wake the SoC from its deepest idle state when data is transmitted to the MBB device over the cellular network.

Power management checklist

System integrators, MBB device vendors, and SoC designers should review the checklist below to ensure that their system power management design is compatible with Windows 8 and Windows 8.1.

- Select a MBB device that implements support for the Windows Mobile Broadband Class Driver and the Mobile Broadband Interface Model (MBIM).

- Windows includes all of the built-in software required to enumerate, describe, communicate with, and power-manage the MBB device. No additional software is required or supported.

  Verify that your MBB device does not require additional driver software or firmware support on the SoC for power management.

- The MBB device hardware must:
  - Report the self-powered and remote-wake capabilities in the USB descriptor for the device.

    - Be capable of using in-band USB resume signaling to wake the SoC when data from the cellular network is ready for processing.
    - Not require any GPIO lines between the MBB device and the SoC to generate wake signaling or enter low-power modes.
    - Not generate spurious USB in-band wakes. The MBB device must wake the SoC only when data matching programmed patterns arrives for the SoC or other enabled wake events occur.

- The MBB device must be reset when the platform enters the S5 (shutdown) system power state. The system integrator must either:
    - Reset the MBB module in system UEFI boot firmware during POST. Note that Windows requires a modern standby platform to complete POST within two seconds.
    - Place the MBB module on a power rail that is turned off when the platform is in the S5 state. The rail can be turned off by a GPIO-controlled switch or by the system PMIC.

- The SoC used in the platform design must be capable of detecting a USB resume signal and waking from its lowest power state.

- System integrators should verify that the power consumption of the MBB device in the radio-off and no-subscription modes meets the previously described requirements.

Wi-Fi power management for modern standby platforms

Article • 08/24/2021

In a Windows hardware platform that implements the modern standby power model, the Wi-Fi device maintains its connection to the network while the platform is in the modern standby power state. Correct power management of the Wi-Fi device—both during modern standby and when the display is turned on—is essential to achieving long battery life.

Wi-Fi is a ubiquitous, medium-range wireless technology that enables high-bandwidth, low-latency communication between computer systems, devices, and the Internet.

Every Windows hardware platform that supports modern standby must be equipped with either a Wi-Fi device or a mobile broadband (MBB) device. Because Wi-Fi integration is more common and widespread, it is expected that the Wi-Fi device will be enabled and connected nearly all of the time.

Overview

A Wi-Fi device in a modern standby platform must support several key power-management features to reduce power consumption by both the device and the platform as a whole.

The first feature—called *power save mode*—allows the Wi-Fi device to reduce its power consumption while it remains connected to the access point. Power save mode reduces power consumption at the cost of increased data transfer latency. Power save mode is expected to always be enabled when the platform is running on battery power, except when low-latency connections are required (for example, for VOIP calls). For more information, see Wi-Fi Auto Power Save Mode.

The second key power-management feature is *pattern-match wake*. This feature allows Windows to arm the Wi-Fi device to wake the System on a Chip (SoC) when the Wi-Fi device detects a network packet that matches a stored pattern. Pattern-match wake is operational only during modern standby. While pattern-match wake is enabled, the Wi-Fi device operates in a very low-power mode and listens for incoming data destined for specific system services or registered applications (for example, push notifications and email). Meanwhile, the other components in the hardware platform are in a low-power state. For more information, see Network Wake-Up Events.

In addition, Wi-Fi devices in a modern standby platform must support the following run-time power-management features:

- Radio on/off state
- Network list offload (NLO)
- ARP/NS offload
- D0 packet coalescing
- Dynamic DTIM management
- A set of Wi-Fi connectivity triggers, which include wake-on-AP-disconnect

This article provides design guidance for Wi-Fi devices in a modern standby platform that are connected to the SoC over the Secure Digital Input/output (I/O) (SDIO) or PCI Express (PCIe) bus, or physically integrated into the SoC itself. Microsoft has not tested Wi-Fi devices that connect via Universal Serial Bus (USB) in modern standby platforms and publishes no design guidance for such devices.

The Wi-Fi device is managed by a Wi-Fi miniport driver, which the Wi-Fi device vendor provides. Windows issues power-management requests (NDIS OIDs) to the device to configure power save mode and pattern-match wake. Windows enables and disables pattern-match wake as the platform enters and exits modern standby. The Wi-Fi miniport driver handles these requests by configuring the Wi-Fi device over the SDIO, PCIe, or proprietary SoC bus.

The Wi-Fi device in a modern standby platform is almost always powered on and is expected to be highly power efficient. When no data is being transferred over the Wi-Fi link and power save mode is enabled, the Wi-Fi device should consume less than 10 milliwatts on average. If the user has set the radio on/off state to *off* by enabling airplane mode or by explicitly turning off Wi-Fi in the Windows **Settings** application, the Wi-Fi device must consume less than 1 milliwatt on average.

Device power-management modes

The Wi-Fi device must support several power-management modes. Each mode is a combination of device activity, network connectivity, and enablement of pattern-match wake.

Active

The Wi-Fi device is connected to the network and is actively transmitting data.

Transition mechanism: The Wi-Fi device hardware autonomously transitions from connected-idle mode to active mode.

Connected idle

The Wi-Fi device is connected to the network, but is not actively transmitting data.

Transition mechanism:

- The Wi-Fi device hardware autonomously transitions from connected-idle mode to active mode.
- The power consumption of the device in the active mode will be a factor of the wireless technology (that is, 802.11a/b/g/n), distance to the access point, quantity of data transmitted, etc.

Connected sleep

The Wi-Fi device is connected to the access point, but the remainder of the platform is in a very low-power state. Pattern-match wake is enabled so that the Wi-Fi device wakes the SoC on a specific set of incoming network packets.

Transition mechanism:

- Before the Wi-Fi device leaves D0, NDIS will send an OID_PM_ADD_WOL_PATTERN request to instruct the Wi-Fi miniport driver to add wake-on-LAN patterns.
- To instruct the Wi-Fi miniport driver to enable pattern-match wake, NDIS will send an OID_PM_PARAMETERS request.
- NDIS will send an OID_PNP_SET_POWER request with an **NDIS_DEVICE_POWER_STATE** value of NdisDeviceStateD2 (for SDIO) or NdisDeviceStateD3 (for PCIe).

Disconnected sleep

The Wi-Fi device is powered but is not connected to an access point, because no preferred access point is within range. The remainder of the platform is in a very low-power state. Pattern-match wake is enabled and the Network Offload List is plumbed to the Wi-Fi device. The Wi-Fi device uses the Network Offload List to periodically scan for preferred networks to connect to.

Transition mechanism:

- The Wi-Fi device uses the network offload list to periodically scan for preferred networks to connect to.
- If a matching network is found during these periodic scans, the Wi-Fi device will wake the SoC.

Radio off

The Wi-Fi device still has power applied, but the radio (RF components) has been powered down.

Transition mechanism:

- While in D0, NDIS will send an OID_DOT11_NIC_POWER_STATE request with a value of FALSE, indicating the radio should be powered off.

Device powered down

The Wi-Fi device has been completely powered down.

Transition mechanism:

- NDIS will send an OID_PNP_SET_POWER request with an **NDIS_DEVICE_POWER_STATE** value of NdisDeviceStateD3.
- If the Wi-Fi device is connected to SDIO or PCIe, the system ACPI firmware will remove power from or reset the Wi-Fi device by using a GPIO line from the SoC to the Wi-Fi device.
- If the Wi-Fi device is integrated into the SoC, the system firmware is responsible for powering off or resetting the Wi-Fi device by using a proprietary mechanism.

Average power consumption and exit latency

The following table shows the expected power consumption and latency when exiting to an active state for each power-management mode. For these figures, assume that the Wi-Fi device is constantly connected to a single access point that has WPA2-Personal security, except in disconnected-sleep mode and power-removed mode.

Device power-management mode	Device power state	Average power consumption	Exit latency to active
Active mode	D0	<= 750 milliwatts	N/A
Connected-idle mode	D0	<= 25 milliwatts	<= 100 milliseconds
Connected-sleep mode	D2 for SDIO; D3 for PCIe	<= 10 milliwatts	<= 300 milliseconds
Disconnected-sleep mode	D2 for SDIO; D3 for PCIe	<= 10 milliwatts	<= 300 milliseconds
Radio-off mode	D0 or D2	<= 1 milliwatt	<= 2 seconds
Power-removed mode (wake disabled)	D3	<= 1 milliwatt	<= 5 seconds

For SDIO-connected devices, D2 is the deepest wakeable device power state for modern standby. For PCIe-connected devices, D3 (more specifically, the D3hot substate) is the deepest wakeable device power state for modern standby.

Windows 8, Windows 8.1, and Windows 10 support modern standby on off-SoC Wi-Fi devices that connect via the SDIO bus. Windows 8.1 additionally supports modern standby on off-SoC Wi-Fi devices that connect via the PCIe bus.

Requirements when a Wi-Fi device shares circuitry

In some Wi-Fi device designs, the Wi-Fi device shares analog and RF circuitry with the Bluetooth and optional FM radio communication devices. In these Wi-Fi device designs, there must not be a power state dependency between the Wi-Fi device and the other integrated communications devices. The Wi-Fi device must have internal power gating circuitry to ensure an average power consumption of less than 1 milliwatt in the radio-off mode.

Software power-management mechanisms

Power management of the Wi-Fi device is primarily based on the NDIS commands that the Wi-Fi miniport driver receives from the Windows networking subsystem. The Wi-Fi miniport driver is responsible for translating these NDIS commands (called OID requests) to device-specific messages to send to the Wi-Fi device over the I/O bus.

Developing an NDIS driver is a complex task whose scope is beyond Wi-Fi device power management. For help in understanding this section, review the Native 802.11 Wireless LAN topic. If you are a Wi-Fi developer, you should become familiar with the set of networking device tests in the Windows Hardware Certification Kit and use these tests as the de facto reference for ensuring that a Wi-Fi device and its Wi-Fi miniport driver are operating correctly.

The Wi-Fi device is expected to always be powered on and connected to a network, except when a modern standby platform is in the S5 (shutdown) system power state. Therefore, correct power management of the Wi-Fi device is critical to system battery life both during modern standby and when the display is turned on. The power-management mode of the Wi-Fi device might change frequently as the device responds to software requests to scan for new networks, or the device itself detects that the access point has been disconnected. The software power-management mechanisms and resulting power consumption described in the following discussion assume that the Wi-Fi device is connected to a single access point that has reasonable signal strength.

The Wi-Fi miniport driver must enable the following key power-management features to ensure that the Wi-Fi device is power-efficient both during modern standby and when the display is turned on:

- Power save mode
- Pattern-match wake

The following paragraphs discuss the requirements for these two features in detail.

In addition to power save mode and pattern-match wake, the Wi-Fi device and the Wi-Fi miniport driver must support several other features. For more information, see Other requirements for Wi-Fi power management.

Power save mode

The 802.11 specification describes a Wi-Fi automatic power save mode (aka *power-save polling*) for Wi-Fi networks. In this mode, the Wi-Fi radio skips polling for a subset of broadcast beacons, thereby reducing the device's average power consumption. Windows requires the Wi-Fi device to enable power save mode when the device is in the active (D0), connected-idle (D0), or connected-sleep (D2 for SDIO; D3 for PCIe) power-management mode. Power save mode is not meaningful when the Wi-Fi device is in the radio-off power-management mode.

The Wi-Fi miniport driver must support and enable power save mode when the Wi-Fi device is in the following power-management modes:

- Active (D0)
- Connected-idle (D0)
- Connected-sleep (D2/D3)

The Wi-Fi miniport driver must report to Windows that it supports auto-negotiation of power save mode by setting the **bAutoPowerSaveMode** member of the **DOT11_EXTSTA_ATTRIBUTES** structure to TRUE. In response to this setting, Windows will send an OID_DOT11_POWER_MGMT_MODE_AUTO_ENABLED request to the Wi-Fi miniport driver. This request instructs the driver to automatically enable power save mode. The Wi-Fi miniport driver must use the capabilities of the remote access point and the Wi-Fi device itself to determine the correct power-saving level to enable.

When automatic power save mode is enabled, Windows may send an OID_DOT11_POWER_MGMT_REQUEST request to the Wi-Fi miniport driver. This message specifies one of four defined power-saving levels. The power-saving level in this message must be treated as a hint to the Wi-Fi miniport driver for the power savings expectation of the operating system or user at the current time. The value

returned by the OID_DOT11_POWER_MGMT_REQUEST request can be used when the driver cannot detect that the remote access point supports power save mode.

Pattern-match wake

Windows requires all networking devices in a modern standby platform to support pattern-match wake. This feature enables the network device to monitor the network connection for interesting packets (by comparing incoming packets to patterns stored in the device) while the SoC or chipset is powered off, and to wake the main processor on the SoC or chipset when a matching pattern is detected. This ability to offload Wi-Fi processing from the main processor is key to achieving low-power operation in modern standby. Before the platform enters modern standby and the Wi-Fi device transitions to the connected-sleep (D2 for SDIO; D3 for PCIe) mode, Windows sends the Wi-Fi miniport driver a collection of interesting patterns that the Wi-Fi device must watch for.

Pattern-match wake is the key enabling feature for modern standby. Pattern-match wake is enabled when the Wi-Fi device is in connected-sleep (D2/D3) mode. Before the Wi-Fi device enters this mode, Windows instructs the device to ignore all network packets except for those that match a specified set of patterns. Windows builds the matching patterns based on the services and applications (for example, push notifications and email) that are currently in use. Offloading pattern matching to the Wi-Fi device allows the rest of the hardware platform either to be turned off or to operate in a low-power mode. Meanwhile, the Wi-Fi device stays turned on, waiting for important incoming network traffic.

To support pattern-match wake, the Wi-Fi miniport driver must first advertise that it supports the connected-sleep (D2/D3) mode and that it is capable of waking the system from the D2/D3 power state. To advertise pattern-match wake support, the Wi-Fi miniport driver must do the following:

- Set the NDIS_PM_WOL_BITMAP_PATTERN bit in the **SupportedWoLPacketPatterns** member of the **NDIS_PM_CAPABILITIES** structure to indicate that the Wi-Fi device supports waking the SoC based on bitmap pattern detection.
- Set the **MinPatternWakeUp** member of the **NDIS_PM_CAPABILITIES** structure to NdisDeviceStateD2 (for SDIO) or NdisDeviceStateD3 (for PCIe) to indicate that the Wi-Fi device can wake the SoC when the device is in the D2/D3 power state and a matching pattern is detected.
- Set the **NumTotalWoLPatterns** member of the **NDIS_PM_CAPABILITIES** structure to a value of 22 decimal (or greater). This value indicates that Windows can

program the Wi-Fi device with up to 22 (or more) unique matching patterns to wake the SoC.

- Set the NDIS_PM_WAKE_PACKET_INDICATION_SUPPORTED bit in the **Flags** member of the **NDIS_PM_CAPABILITIES** structure. This flag indicates that the Wi-Fi device supports storage and retrieval of the packet that caused the Wi-Fi device to wake the SoC.

When Windows first detects that a system service or hosted application requires pattern-match wake support, it will send the OID_PM_ADD_WOL_PATTERN request to the Wi-Fi miniport driver to specify the pattern to be matched. This request is sent well before the Wi-Fi device is transitioned to the connected-sleep (D2/D3) mode.

Entering connected standby with pattern-match wake enabled

When the hardware platform enters modern standby, Windows will transition the Wi-Fi device to the connected-sleep (D2 for SDIO; D3 for PCIe) mode by sending an OID_PNP_SET_POWER request that specifies a device power state of NdisDeviceStateD2 (for SDIO) or NdisDeviceStateD3 (for PCIe). In response to this request, the Wi-Fi miniport driver must do the following:

1. Stop sending any new I/O requests to the underlying bus driver or hardware. If the Wi-Fi device is outside the SoC and connected via SDIO, the underlying bus driver will be the Windows SD bus driver, Sdbus.sys. If the Wi-Fi device is outside the SoC and connected via PCIe, the underlying bus driver will be the Windows PCI bus driver, Pci.sys. If the Wi-Fi device is integrated into the SoC, the underlying bus driver will be provided by the SoC vendor.

2. Wait for all in-flight I/O requests, and complete all queued I/O requests.

3. Program the Wi-Fi device for all configured wake patterns and wake events.

4. Transition the Wi-Fi device to the correct low-power state (D2/D3) for connected-sleep mode.

5. If the Wi-Fi device is outside the SoC and attached via an SDIO bus:

    a. Configure the Wi-Fi device to wake the SoC using the out-of-band GPIO wake interrupt. (For more information, see Supported hardware power configurations.)

    b. Mask the Wi-Fi device interrupts and cancel all related timers. The Wi-Fi miniport driver must set the **SDP_FUNCTION_INT_ENABLE** property to FALSE by calling the **SdBusSubmitRequest** routine.

c. Instruct the SD bus driver to stop forwarding Wi-Fi device interrupts. The Wi-Fi miniport driver must set the **SDP_SET_CARD_INTERRUPT_FORWARD** property to FALSE by calling the **SdBusSubmitRequest** routine.

d. Complete the OID_PNP_SET_POWER request for the NdisDeviceStateD2 state.

6. If the Wi-Fi device is outside the SoC and attached via the PCIe bus:

a. Configure the Wi-Fi device to wake the system in accordance with the PCIe standard (using PM_PME messages; this is discussed in further detail in the following section.)

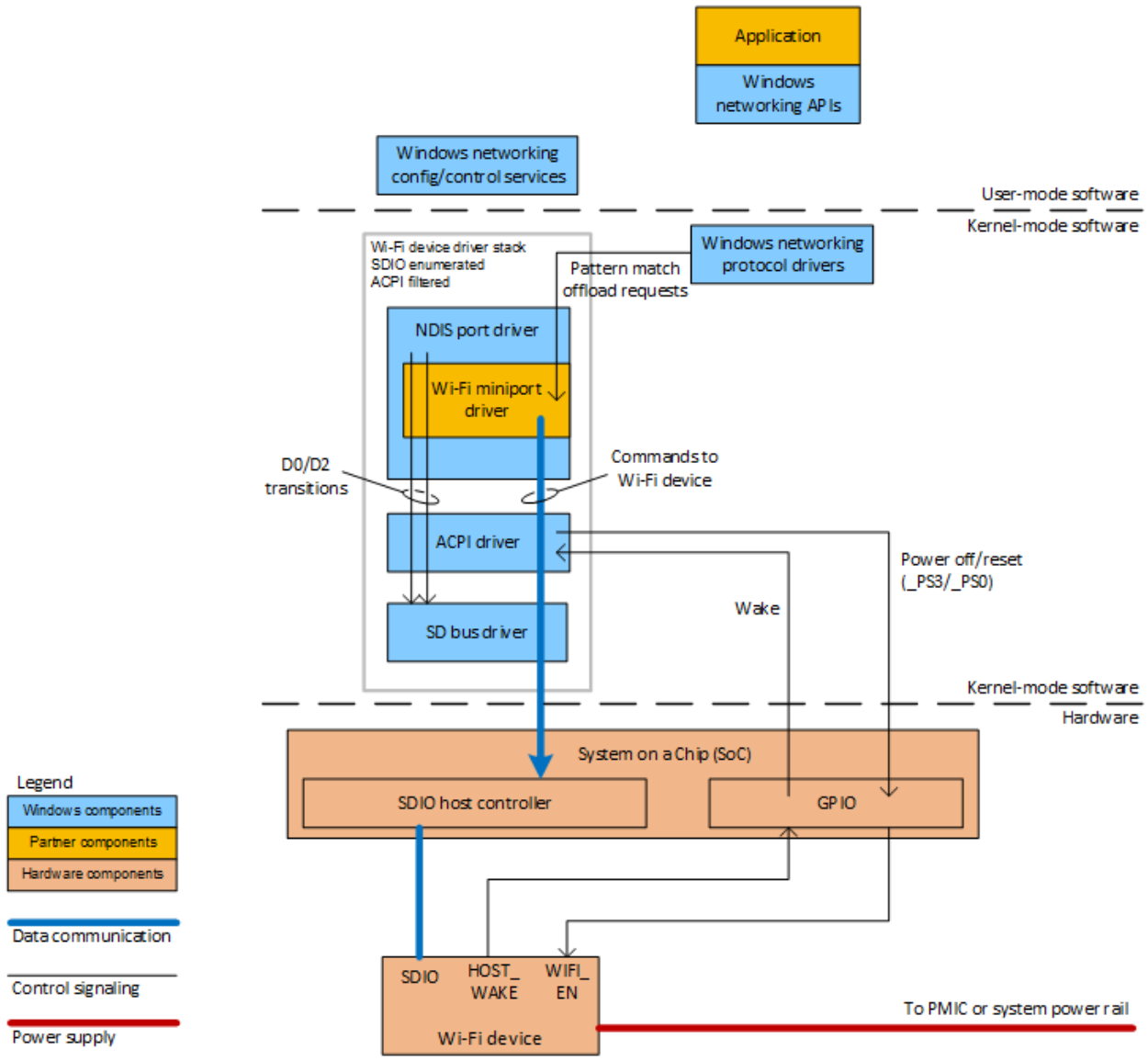
b. Mask the Wi-Fi device interrupts. The Wi-Fi miniport driver must disconnect interrupts by calling the **IoDisconnectInterruptEx** routine.

c. Complete the OID_PNP_SET_POWER request for the NdisDeviceStateD3 state.

When the Wi-Fi device is in the connected-sleep mode, the device is expected to maintain its connection to the associated Wi-Fi network, to check for matching wake patterns, and to wake the SoC when network state changes occur. For example, the Wi-Fi device must be able to wake the SoC if the access point is suddenly disassociated.

The following figure shows the configuration of software and hardware components to support a Wi-Fi device that is external to the SoC and that communicates with the SoC

through an SDIO bus.



Exiting modern standby for a matching pattern

When the Wi-Fi device detects a packet with a matching pattern, it must first save this packet in local non-volatile storage for later retrieval by the Wi-Fi miniport driver. Windows will have the driver retrieve information about the packet so that the appropriate system service or application can be notified of the incoming packet.

Next, the Wi-Fi device must generate an interrupt to wake the SoC.

If the Wi-Fi device is located outside of the SoC on an SDIO port, the interrupt should be delivered through the out-of-band GPIO interrupt and cause the SoC to wake. The ACPI subsystem then determines that the GPIO interrupt for the Wi-Fi device has caused the SoC to wake, and will respond by completing a wait-wake IRP (an **IRP_MN_WAIT_WAKE** request) that the NDIS port driver automatically left pending in the ACPI driver on behalf of the Wi-Fi miniport driver.

If the Wi-Fi device is located outside the SoC on a PCIe bus, and the ACPI firmware grants control of native PCIe features to the operating system, wake events are handled according to the PCIe specification, by using the following steps:

1. When the Wi-Fi device goes to a low-power state, the NDIS port driver automatically sends a wait-wake IRP down the Wi-Fi driver stack to the inbox PCI bus driver, Pci.sys (not the ACPI driver, Acpi.sys), which marks the IRP as pending in anticipation of a future wake event.

2. When the Wi-Fi device needs to wake, it should generate a WAKE notification to the root of the PCIe hierarchy. To do so, the device sends an in-band PM_PME message over the PCIe bus.

3. The wake notification is routed to the root port of the PCIe hierarchy above the Wi-Fi device. The root port will be in the D0 power state and is expected to generate a PME interrupt to the inbox PCI bus driver, Pci.sys.

4. The PCI bus driver on the root port brings the entire device hierarchy beneath that root port back to D0.

5. The PCI bus driver examines the PME_Status bit of each device in the hierarchy to determine if it issued a WAKE notification. If the Wi-Fi device's PMEStatus bit is set, the PCI bus driver then completes any outstanding wait-wake IRPs that are pending for that device, which causes the IRP pended in step 1 to return to NDIS.

For more information, see Low Power for Wake on LAN.

If the Wi-Fi device is integrated into the SoC, the operation of the device is similar to that described for the off-SoC, SDIO-port implementation. The SoC vendor must describe the device in the DSDT table in the ACPI firmware. The ACPI subsystem will then enumerate the Wi-Fi device and complete the wait-wake IRP. We recommend that system integrators who use SoCs that have integrated Wi-Fi devices work closely with the SoC vendor on Wi-Fi device integration and driver development.

When the wait-wake IRP is completed, NDIS will first send a D0 IRP (an **IRP_MN_SET_POWER** request of type **DevicePowerState**) down the stack to the bus driver. Before completing the D0 IRP, NDIS will send the Wi-Fi miniport driver an OID_PNP_SET_POWER request with a target state of NdisDeviceStateD0. In response, the Wi-Fi miniport driver must do the following:

1. Inspect the Wi-Fi device hardware to determine the reason for the wake interrupt. The Wi-Fi miniport driver must notify NDIS of the reason for the wake interrupt by generating an **NDIS_STATUS_PM_WAKE_REASON** status event. If the wake interrupt is caused by an incoming network packet, the driver must pass this

packet to NDIS through the regular NDIS data path before the driver passing any other received packets to NDIS.

2. Restore any register state or other hardware context needed to complete the Wi-Fi device's transition to connected-idle (D0) mode.

3. If the Wi-Fi device is external to the SoC and connected via SDIO, the Wi-Fi miniport driver must:

   a. Instruct the SD bus driver to forward interrupts to the Wi-Fi miniport driver. The Wi-Fi miniport driver must set the **SDP_SET_CARD_INTERRUPT_FORWARD** property to TRUE by calling the **SdBusSubmitRequest** routine.

   b. Mask the Wi-Fi device interrupts. The Wi-Fi miniport driver must set the **SDP_FUNCTION_INT_ENABLE** property to TRUE by calling the **SdBusSubmitRequest** routine.

4. Complete the **IRP_MN_SET_POWER** request.

Pattern-match wake and the ability to wake the SoC from a low-power idle state are essential to maintaining network connectivity during modern standby. Wi-Fi driver developers should work closely with Microsoft to develop a Wi-Fi miniport driver that implements these required features.

Other requirements for Wi-Fi power management

In addition to power save mode and pattern-match wake, the Wi-Fi device and miniport driver must support several other run-time power-management features for Wi-Fi devices in modern standby platforms. These features are essential for reducing power consumption in the Wi-Fi device and overall hardware platform. As previously discussed, the Wi-Fi device and miniport driver must support the following features:

- Radio on/off state
- Network list offload (NLO)
- ARP/NS offload
- D0 packet coalescing
- Dynamic DTIM configuration
- Wi-Fi connectivity wake triggers, which include wake-on-AP-disconnect

Radio on/off state

The Wi-Fi miniport driver must respond to Windows requests to change the radio on/off state for the Wi-Fi device. The user can change the radio on/off state to *off* by enabling

airplane mode or by explicitly turning off Wi-Fi in the Windows **Settings** application.

When the user changes the radio on/off state of the Wi-Fi device to *off*, Windows sends the Wi-Fi miniport driver an OID_DOT11_NIC_POWER_STATE request with a value of FALSE. In response, the Wi-Fi miniport driver must use an in-band message to remove power from the radio (RF subsystem) in the Wi-Fi device and place the device in a power state that consumes no more than 1 milliwatt on average.

When the user changes the radio on/off state of the Wi-Fi device to *on*, Windows sends the Wi-Fi miniport driver an OID_DOT11_NIC_POWER_STATE request with a value of TRUE. In response, the Wi-Fi miniport driver must use an in-band message to apply power to the radio in the Wi-Fi device and return the device to the connected-idle power mode.

When the radio on/off state of the Wi-Fi device is set to *off*, the Wi-Fi device can be in either the D0 or D2 power state. NDIS will always transition the Wi-Fi device to the D0 power state before notifying the Wi-Fi miniport driver of a change in the radio on/off state. If the user has previously set the radio on/off state to *off*, the Wi-Fi miniport driver does not need to take any specific action when the hardware platform enters modern standby. If the radio on/off state is set to *off*, the radio will remain off in modern standby, and, after the system exits modern standby, the radio on/off state will still be set to *off* until the user changes the state to *on*.

Network list offload (NLO)

The network list offload (NLO) feature enables Wi-Fi profile information to be copied to local storage in the Wi-Fi device. Instead of scanning all available channels for all access points, the Wi-Fi device uses this profile information to scan a limited number of channels for preferred networks, thereby reducing the power consumed by the Wi-Fi device. The Wi-Fi device and Wi-Fi miniport driver in a modern standby platform must support network list offload. This support includes handling OID_DOT11_OFFLOAD_NETWORK_LIST requests.

To advertise NLO capability, the Wi-Fi miniport driver must set the NDIS_WLAN_WAKE_ON_NLO_DISCOVERY_SUPPORTED flag in the **MediaSpecificWakeUpEvents** member of the **NDIS_PM_CAPABILITIES** structure.

ARP/NS offload

The Wi-Fi miniport driver and Wi-Fi device must support ARP/NS offload, which allows the Wi-Fi device to autonomously respond to common network requests. The ARP/NS offload feature avoids waking the SoC for common network requests that have simple

and predictable responses. To indicate support for ARP/NS offload, the Wi-Fi miniport driver must set the NDIS_PM_PROTOCOL_OFFLOAD_ARP_SUPPORTED and NDIS_PM_PROTOCOL_OFFLOAD_NS_SUPPORTED flags in the **SupportedProtocolOffloads** member of the **NDIS_PM_CAPABILITIES** structure.

Additionally, the Wi-Fi miniport driver and Wi-Fi device must support at least one IPv4 ARP offload address and at least two DIPv6 NS offload addresses. The Wi-Fi miniport driver must set the **NumArpOffloadIPv4Addresses** member of the **NDIS_PM_CAPABILITIES** structure to a value of 1 or greater. The Wi-Fi miniport driver must set the value of the **NumNSOffloadIPv6Addresses** member of the **NDIS_PM_CAPABILITIES** structure to a value of 2 or greater. Windows uses the OID_PM_ADD_PROTOCOL_OFFLOAD request to supply the ARP and NS offload addresses to the Wi-Fi miniport driver.

D0 packet coalescing

The Wi-Fi device and Wi-Fi miniport driver must support D0 packet coalescing, which is a feature that enables the Wi-Fi device to batch common, low-priority network packets for batched retrieval by the SoC. This feature reduces the overall quantity and frequency of core chipset interrupts so that the SoC can stay longer in lower-power states, thereby extending battery life. The Wi-Fi miniport driver advertises support for D0 packet coalescing by setting several values in the **NDIS_RECEIVE_FILTER_CAPABILITIES** structure.

The following list of members summarizes the D0 packet-coalescing capabilities that the Wi-Fi device is required to support and the Wi-Fi miniport driver is required to report. The Wi-Fi miniport driver must specify these capabilities in the **NDIS_RECEIVE_FILTER_CAPABILITIES** structure. For each member, the list shows the minimum value required for a particular member of this structure. NDIS will send OID_RECEIVE_FILTER_SET_FILTER requests to the Wi-Fi miniport driver to set the packet-coalescing filters. The packet-coalescing filters indicate which packets are required to be batched in a coalescing buffer on the Wi-Fi device.
SupportedQueueProperties

NDIS_RECEIVE_FILTER_PACKET_COALESCING_SUPPORTED_ON_DEFAULT_QUEUE

> **Note** This flag must always be present in the **HardwareReceiveFilterCapabilities** to indicate the hardware capability. This flag must be present in the **CurrentReceiveFilterCapabilities** if and only if the ***PacketCoalescing** advanced keyword is nonzero.

EnabledFilterTypes, minimum value:

NDIS_RECEIVE_FILTER_PACKET_COALESCING_FILTERS_ENABLED
SupportedFilterTests, minimum values:

NDIS_RECEIVE_FILTER_TEST_HEADER_FIELD_EQUAL_SUPPORTED
NDIS_RECEIVE_FILTER_TEST_HEADER_FIELD_NOT_EQUAL_SUPPORTED
NDIS_RECEIVE_FILTER_TEST_HEADER_FIELD_MASK_EQUAL_SUPPORTED
SupportedHeaders, minimum values:

NDIS_RECEIVE_FILTER_MAC_HEADER_SUPPORTED
NDIS_RECEIVE_FILTER_ARP_HEADER_SUPPORTED
NDIS_RECEIVE_FILTER_IPV4_HEADER_SUPPORTED
NDIS_RECEIVE_FILTER_IPV6_HEADER_SUPPORTED
NDIS_RECEIVE_FILTER_UDP_HEADER_SUPPORTED
SupportedMacHeaderFields, minimum values:

NDIS_RECEIVE_FILTER_MAC_HEADER_DEST_ADDR_SUPPORTED
NDIS_RECEIVE_FILTER_MAC_HEADER_PROTOCOL_SUPPORTED
NDIS_RECEIVE_FILTER_MAC_HEADER_PACKET_TYPE_SUPPORTED
SupportedARPHeaderFields, minimum values:

NDIS_RECEIVE_FILTER_ARP_HEADER_OPERATION_SUPPORTED
NDIS_RECEIVE_FILTER_ARP_HEADER_SPA_SUPPORTED
NDIS_RECEIVE_FILTER_ARP_HEADER_TPA_SUPPORTED
SupportedIPv4HeaderFields, minimum value:

NDIS_RECEIVE_FILTER_IPV4_HEADER_PROTOCOL_SUPPORTED
SupportedIPv6HeaderFields, minimum value:

NDIS_RECEIVE_FILTER_IPV6_HEADER_PROTOCOL_SUPPORTED
SupportedUdpHeaderFields, minimum value:

NDIS_RECEIVE_FILTER_UDP_HEADER_DEST_PORT_SUPPORTED
MaxFieldTestsPerPacketCoalescingFilter, minimum value:

5
MaxPacketCoalescingFilters, minimum value:

10

NDIS sends OID_RECEIVE_FILTER_CLEAR_FILTER requests to clear previously set packet-coalescing filters when these filters are no longer required. NDIS might set and clear individual packet-coalescing filters dynamically. The Wi-Fi miniport driver is required to keep track of the current set of packet-coalescing filters at all times, but can apply

coalescing only when the Wi-Fi device is in the D0 power state (and is in either active mode or connected-idle mode).

Dynamic DTIM configuration

The Wi-Fi miniport driver must support dynamic configuration of the *delivery traffic indication message* (DTIM) interval. Dynamically adjusting the DTIM interval to a larger value during connected-sleep (D2 for SDIO; D3 for PCIe) mode allows the Wi-Fi device to consume very little power while continuously maintaining a connection to the access point. The additional latency that results from the longer DTIM interval is not critical in connected-sleep (D2/D3) mode.

The Wi-Fi miniport driver should support dynamic DTIM intervals by implementing the following guidance:

- The Wi-Fi device (in STA mode) must advertise a listen interval value of 10 to the access point. This value will force the access point to cache data for the Wi-Fi client for 10 beacon intervals.

- To prepare the Wi-Fi device to enter connected-sleep (D2/D3) mode, the driver must change the length of the DTIM interval to approximately 500 milliseconds. The DTIM value to specify depends on the value of the normal *traffic indication message* (TIM). For example, if the TIM is currently 100 milliseconds, the Wi-Fi device should use a DTIM value of 5 (for an interval of 500 milliseconds). If the TIM is currently 300 milliseconds, the Wi-Fi device should use a DTIM value of 2 (for an interval of 600 milliseconds).

When the Wi-Fi device transitions back to the connected-idle (D0) mode, the Wi-Fi device must revert to the original DTIM of that was negotiated with the access point.

Wi-Fi connectivity wake triggers

When the Wi-Fi device is connected to a network during modern standby, the Wi-Fi device and the Wi-Fi miniport driver must also support waking the SoC based on a set of Wi-Fi connectivity wake triggers. The requirements are as follows:

- Perform GTK (WPA/WPA2) and IGTK refresh (WPA2) while in modern standby.
- Support waking on GTK and IGTK handshake errors.
- Support waking when an 802.1x EAP-Request/Identity packet is received.
- Support waking when a four-way handshake request is received.
- Support wake-on-association-lost with current access point.

Waking the SoC on these Wi-Fi specific events allows Windows to be notified when Wi-Fi connectivity is in jeopardy or when the Wi-Fi device loses connectivity to the associated access point. In response, Windows might instruct the Wi-Fi miniport driver and device to connect to an alternate Wi-Fi network. Or, Windows might instead use the mobile broadband (MBB) radio to establish a connection. The Wi-Fi miniport driver must specify each of these wake trigger capabilities (for example, by setting the NDIS_WLAN_WAKE_ON_AP_ASSOCIATION_LOST_SUPPORTED flag) in the **SupportedWakeUpEvents** member of the **NDIS_PM_CAPABILITIES** structure.

Supported hardware power configurations

Windows supports three hardware power-management configurations for the Wi-Fi device in a modern standby platform. The Wi-Fi device either must be located outside the SoC and attached via SDIO or PCIe, or must be physically integrated into the SoC chip and attached via a proprietary internal bus.

The system designer must not use a USB bus connection to integrate a Wi-Fi device in a modern standby platform.

Outside the SoC, attached via SDIO

In this configuration, which is shown in the preceding block diagram, the Wi-Fi device is located outside of the SoC and is connected to the SoC through an SDIO bus. The Wi-Fi device might have one or more pins connected to GPIO pins on the SoC. The Wi-Fi device must be either attached to a power rail that is always powered on, or directly connected to the power-management IC (PMIC).

The Wi-Fi device must be able to generate a wake interrupt over an out-of-band signal line routed from an interrupt output pin on the Wi-Fi device to a GPIO pin on the SoC. The interrupt line from the device must be connected to an always-on GPIO pin on the SoC so that the Wi-Fi device's wake signal can wake the SoC from its lowest power state during modern standby. The system integrator must use the ACPI GpioInt macro to describe the GPIO interrupt connection under the Wi-Fi device in the ACPI namespace. The **Shared** field of the GpioInt macro must be set to ExclusiveAndWake to indicate that the GPIO interrupt is capable of waking the system from sleep. The ExclusiveAndWake setting enables the ACPI driver to automatically complete the wait-wake IRP for the Wi-Fi device, and to return the device to connected-idle (D0) mode if the device is in connected-sleep (D2) mode when it interrupts the SoC. The ACPI firmware must indicate that the Wi-Fi device can wake the platform from connected-sleep (D2) mode. To do so, the system integrator must include both _S4W and _S0W objects in the ACPI namespace under the Wi-Fi device, as shown in the following example:

```
    Name(_S0W, 2)
    Name(_S4W, 2)
```

The value 2 in this example indicates the D2 device power state.

As an option, the Wi-Fi device can have an input pin that accepts an enable or reset signal from a GPIO output pin on the SoC. The GPIO pin is controlled by system firmware. The GPIO pin can be switched to completely power down or reset the Wi-Fi device. If such a GPIO pin is used to enable or reset the Wi-Fi device, the system integrator must expose this pin in a GPIO operation region. The system integrator must also include _PS0 and _PS3 control methods under the Wi-Fi device in the ACPI namespace.

The implementation of the _PS3 method must power down or reset the Wi-Fi device by changing the state of the GPIO output pin through the GPIO operation region. If the Wi-Fi device has specific reset timing requirements, the system integrator must implement these requirements in ACPI firmware by using the ACPI Sleep method to delay execution or to hold the reset line to the required signal level for some specified time.

The implementation of the _PS0 control method must reliably apply power to the Wi-Fi device and make the device operational.

Outside the SoC, attached via PCIe

In this configuration, the Wi-Fi device is located outside of the SoC and attached to the PCIe bus. The device is in the D3 device power state when it operates in connected-sleep mode or disconnected-sleep mode. While in D3, the device remains in the D3hot substate and does not enter the D3cold substate. The Wi-Fi device must be attached to a system power rail that is always powered on or is directly connected to the system power management IC (PMIC).

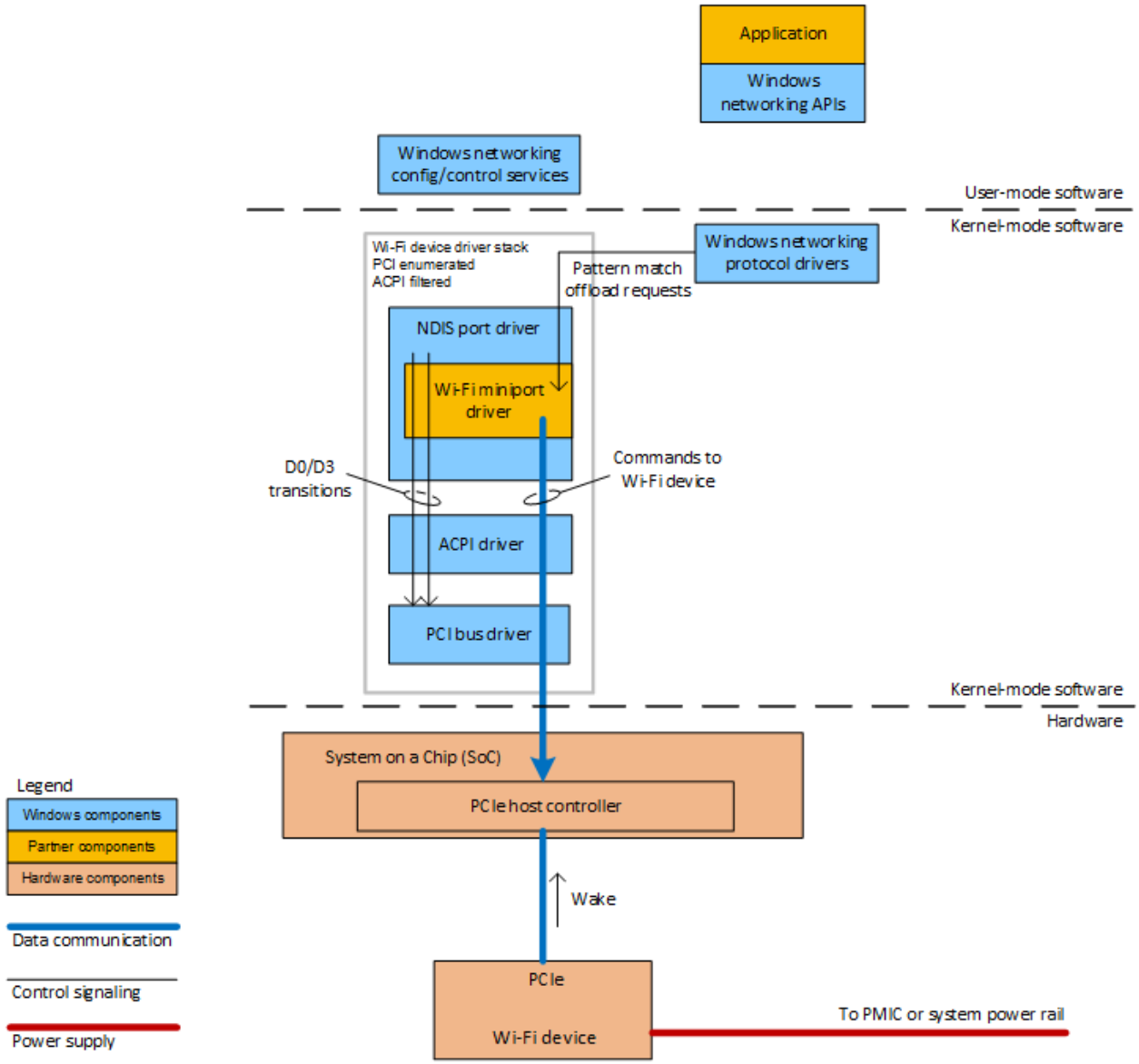
> **Note** The Wi-Fi hardware should use PCI architectural means of generating a wake event (PME).

When in D3, the device must be able to signal a wake event by sending a PM_PME message that propagates in-band over the PCIe bus. The wake event will generate an interrupt from the PCIe root port, and this interrupt will be handled by the inbox PCI bus driver, Pci.sys.

To grant the operating system control over native PCIe features, the ACPI firmware must include an _OSC control method in the ACPI namespace. In addition, the ACPI namespace must include an _S0W object to indicate that the Wi-Fi device can wake the platform from connected-sleep mode or disconnected-sleep mode. This object must be located under the Wi-Fi device in the ACPI namespace, and declared as shown in the following example:

```
Name(_S0W, 3)
```

The following figure shows the configuration of software and hardware components to support a Wi-Fi device that is external to the SoC and that communicates with the SoC through the PCIe bus.



Integrated into the SoC

If the Wi-Fi device is integrated into the SoC, tight coupling between the Wi-Fi miniport driver and a proprietary SoC bus driver is required. The implementation of this driver is outside of the scope of this document. However, the Wi-Fi miniport driver must implement all of the features described in Software power-management mechanisms.

The system integrator should contact the SoC vendor for ACPI implementation details for Wi-Fi devices that are directly integrated into the SoC.

Testing and validation

Testing and validation of the Wi-Fi device should focus on directly measuring power consumption and verifying that pattern-match wake works correctly.

The direct measurement of device power consumption is a critical part of testing and validation for a Wi-Fi device. Because the Wi-Fi device is almost always powered on when the platform is not in the S5 (shutown) system power state, the system integrator must test and validate Wi-Fi power consumption in a number of different system scenarios and configurations. We recommend directly measuring the power consumption of the Wi-Fi device for the two types of connections shown in the following table.

Scenario	Target average power	Comments
Airplane mode, radio off • Screen on, D0 state	<= 1 milliwatt	The radio might be turned off both when the display is on and during modern standby. When the radio is off, the power consumption must be no more than 1 milliwatt on average for both D0 and D2.
Screen on, idle • Power save mode enabled on automatic • No active data transfers • DTIM configured to access point support • WPA2-Personal security	<= 10 milliwatts	Testing should be performed with the Wi-Fi device connected to a variety of consumer and commercial Wi-Fi access points.

Scenario	Target average power	Comments
Connected sleep • Screen off, pattern-match wake enabled, D2 state • DTIM configured to 5 • WPA2-Personal security	<= 10 milliwatts	

Power-management checklist

System integrators and SoC vendors should use the checklist below to verify that their Wi-Fi device and Wi-Fi miniport power-management design are compatible with Windows 8 and Windows 8.1.

Note The Windows Hardware Certification Kit includes an extensive set of Wi-Fi driver tests to help ensure the Wi-Fi device is compatible with Windows 8 and Windows 8.1. Wi-Fi device vendors and Wi-Fi miniport driver developers are encouraged to review the Windows Hardware Certification Kit tests and use them to validate their driver implementation as early as possible in the design cycle.

- The Wi-Fi device vendor must develop a driver that does the following:
  - Supports pattern-match wake.
  - Supports power save mode.
  - Supports radio on/off.
  - Supports network list offload (NLO).
  - Supports D0 packet coalescing.
  - Supports wake-on-AP-disconnect and additional Wi-Fi wake triggers.
  - Supports ARP/NS offload.
  - Supports dynamic DTIM configuration.
  - Passes all applicable networking tests in the Windows Hardware Certification Kit.

- If the Wi-Fi device is located outside of the SoC and attached via SDIO, the system integrator must develop the hardware and firmware for the Wi-Fi device to do the following:

  - Route the Wi-Fi wake interrupt from the Wi-Fi device to an always-on GPIO pin on the SoC.

  - Include the GPIOInt macro under the Wi-Fi device in the ACPI namespace with the Shared field set to ExclusiveAndWake.

- Include the _S4W object under the Wi-Fi device in the ACPI namespace with a deepest wake state of 0x2 for D2. Specify as "Name(_S4W, 2)".
- Include the _S0W object under the Wi-Fi device in the ACPI namespace with a deepest wake state of 0x2 for D2. Specify as "Name(_S0W, 2)".
- Route any power-enable or reset signal from a GPIO output pin on the SoC to the Wi-Fi device.
- Describe the optional power-enable or reset GPIO output pin in a GPIO operation region.
- Provide _PS3 and _PS0 control methods and/or _PR*n* power resource references. These entries should be under the Wi-Fi device in the ACPI namespace.
- Implement the _PS3 method or power resource methods to remove power from the Wi-Fi device.
- Implement the _PS0 method or power resource methods to restore power to the Wi-Fi device.
- Include any device-specific power-on or reset sequencing timing in the implementation of the _PS3 and _PS0 methods. ACPI method execution can be delayed by using the Sleep method.

- If the Wi-Fi device is located outside of the SoC and attached via PCIe, the system integrator must develop the hardware and firmware for the Wi-Fi device to do the following:
  - Include an _OSC control method in the ACPI namespace to grant the operating system control over native PCIe features.

    Include the _S4W object under the Wi-Fi device in the ACPI namespace with a deepest wake state of 0x3 for D3. Specify as "Name(_S4W, 3)".

- The system integrator must test and validate correct power management of the Wi-Fi device by doing the following:
  - Verify that the average power consumption of the Wi-Fi device in radio-off mode is less than 1 milliwatt.
  - Verify that the average power consumption of the Wi-Fi device is less than 10 milliwatts when the device is in connected-sleep (D2 for SDIO; D3 for PCIe) mode or connected-idle (D0) mode and is not actively transmitting any data.

- Verify that the Wi-Fi miniport correctly supports pattern-match wake and is capable of waking the SoC from its deepest idle state when a matching pattern is detected.
- Verify that the Wi-Fi device can wake the SoC from its deepest idle state when the device loses its connection to the associated access point.
- Verify that the Wi-Fi device does not generate spurious wakes to the SoC.
- Use the tests provided in the Windows Hardware Certification Kit to verify that the Wi-Fi device correctly implements network list offload (NLO), ARP/NS offload, and D0 packet coalescing.

Near-field proximity (NFP) power management for Modern Standby platforms

Article • 11/05/2020

A near-field proximity (NFP) device is a short-range radio transmitter/receiver device that enables a Windows PC to associate and pair with other NFP-equipped computers.

An NFP device is exposed to the Windows operating system through the GUID_DEVINTERFACE_NFP Plug and Play (PnP) interface. A third-party driver, provided by the NFP device vendor, is responsible for implementing the GUI_DEVINTERFACE_NFP interface and power-managing the physical NFP device. The NFP device is typically connected to the I2C interface pins of the System on a Chip (SoC) or core silicon.

Generally, the NFP device can enter a low-power mode (standby mode or power-removed mode) when either no software clients have proximity publications or subscriptions, or all publications and subscriptions are disabled. From a driver development point of view, this condition occurs when there are no handles outstanding on the device. The NFP device should be placed in either standby mode or power-removed mode when no handles to the device are open and enabled.

When the platform enters Modern Standby and the display is turned off, the NFP device must enter either standby mode or power-removed power mode. If an NFP device's power consumption in standby mode is greater than one milliwatt on average, then the device must enter a 0-watt, power-removed mode.

The NFP device driver is not directly notified when the platform enters and exits Modern Standby. Instead, Windows notifies the driver indirectly by disabling the handles to the driver on entry to Modern Standby, and enabling these same handles on exit from Modern Standby. After the platform enters Modern Standby, Windows sends a disable notification to each open handle that is not already disabled. When the platform exits Modern Standby, Windows sends an enable notification to each handle that it previously disabled. An open handle that was already disabled before Modern Standby was entered stays disabled after the exit from Modern Standby.

Device power management modes

Starting with Windows 8, an NFP device for a Modern Standby platform is expected to have three power modes — *active*, *idle*, and *standby* — in addition to a power-removed

mode (in which all power inputs to the NFP device have been switched off by a hardware mechanism external to the device). Contact the hardware vendor for an NFP device for details about the power management modes supported by the device.

The following table describes the device power modes for an NFP device.

Device power mode	Description	Average power consumption	Exit latency to active	Transition mechanism
Active	NFP device is actively communicating with another NFP device on a nearby computer.	< 50 milliwatts	N/A	N/A
Idle	NFP device is waiting for another computer with NFP to come nearby. The driver for the NFP device has open publications and subscriptions.	< 5 milliwatts	< 10 milliseconds (typical)	Hardware autonomous
Standby	The NFP device has power applied, but there are no outstanding publications or subscriptions (no open handles).	< 1 milliwatt (device-dependent)	< 50 milliseconds (typical)	Driver command or GPIO pin from the SoC or core silicon.
Power-removed	The NFP device has no power applied because the platform is not powered on or an external entity has switched off power.	0 watts	< 100 milliseconds	External entity removes power or applies power via ACPI firmware in response to a D3 IRP.

ⓘ Note

- In the preceding table, the term *standby* refers to a device power mode that is distinct from *Modern Standby*, which is a platform-wide power state.
- In the preceding table, the term *D3 IRP* refers to an IRP_MN_SET_POWER request of type **DevicePowerState**.

The specified power consumption of an NFP device in standby mode will vary by part number and manufacturer. System designers should work with the device vendor to understand the device-specific standby power consumption. If the power consumption in standby mode is more than one milliwatt, the NFP device should be attached to a switchable power rail so that the device can be put in power-removed mode during Modern Standby.

Software power management mechanism

Windows relies on the NFP device vendor to provide a third-party driver that power-manages the NFP device. Windows expects this driver to monitor the current number of open handles to the NFP device driver and the state of these handles to determine when to put the NFP device in a low-power mode.

We recommend that NFP driver developers use the User-Mode Driver Framework (UMDF) to develop their drivers. UMDF includes a number of features that make development of device run-time power management easier.

D3 support

NFP driver developers are encouraged to put the NFP device in a D3 state when all handles to the device have been closed or all open handles to the device have received the **IOCTL_NFP_DISABLE** notification. All open handles will receive the **IOCTL_NFP_DISABLE** notification after the platform enters Modern Standby and the user desktop is locked.

Even if the NFP device can attain a power consumption of less than one milliwatt in idle power mode, a transition to D3 is still recommended when all handles are closed or have received the **IOCTL_NFP_DISABLE** notification. Performing a transition to D3 allows all drivers in the driver stack for the NFP device to be notified that the hardware is entering a low-power mode. Additionally, a transition to D3 can be detected by instrumentation that is built into Windows. The Windows Performance Toolkit and other performance diagnostic tools can be used to observe this transition. This built-in diagnostic support reduces the cost for a system integrator to verify that the NFP device is IOCTL_NFP_DISABLE correctly performing power management.

For some buses, the devices on the bus must transition to D3 before the bus controller can enter a low-power state. For an I2C-connected NFP device, there is no such dependence on the endpoint NFP device entering D3 in order for the I2C bus controller to power down.

However, if Modern Standby platforms in the future use Universal Serial Bus (USB) to connect to NFP devices, an NFP device must be able to enter D3 so that the USB host controller can power down and the SoC or core silicon can enter a low-power idle state during Modern Standby. NFP device driver developers are encouraged to transition their devices to D3 when they're ready to enter a low-power mode, regardless of the underlying bus, to enable future driver re-use and easy power management diagnostics.

If, to enter standby power mode, the NFP device requires a command to be sent from the NFP device driver over the communications bus, this command should be sent as part of the transition to D3 in the driver.

If the NFP device requires a GPIO pin from the SoC to be toggled to enter standby power mode, a transition to D3 is required. Initiating a transition to D3 notifies the Windows ACPI driver, Acpi.sys, which executes the _PS3 control method implemented under the NFP device in the ACPI namespace. The _PS3 method can toggle the GPIO line through a GPIO operation region. To make the NFP device driver more portable, any platform-specific delays or other timing constraints should be implemented in the platform firmware so that the NFP device driver does not need to be customized for a specific platform.

To enable the transition of the NFP device to D3 when the device is idle and the platform is in the S0 (working) system power state, a UMDF driver calls the IWDFDevice2::AssignS0IdleSettingsEx method. In this call, the driver supplies a pointer to a WUDF_DEVICE_POWER_POLICY_IDLE_SETTINGS structure as the input parameter, and sets the DxState member of this structure to PowerDeviceD3.

If the power consumption of the NFP device in standby mode is greater than one milliwatt, the device must be put in power-removed mode when all handles are closed or all handles have been sent the **IOCTL_NFP_DISABLE** notification. In order for the underlying firmware and hardware to remove power from the NFP device, the device must already have entered D3 (that is, D3hot) and the NFP device driver must previously have enabled the transition to D3cold. Enabling D3cold implies that the driver for the NFP device saves the device's hardware state before the transition to D3 so that all power can be removed from the device. The device must be able to resume normal operation at a later time after the device is powered on (that is, the device exits D3cold) and the driver restores the device's hardware state. To enable D3cold, a UMDF driver calls the IWDFDevice2::AssignS0IdleSettingsEx method. In this call, the driver supplies a

pointer to a WUDF_DEVICE_POWER_POLICY_IDLE_SETTINGS structure, and sets the ExcludeD3Cold member of this structure to WdfFalse.

For more information, see Supporting D3cold in a Driver.

Radio manager

If the NFP device uses an RF (radio) mechanism to detect proximity events, the NFP device vendor must supply a radio manager in addition to an NFP device driver. The radio manager is a COM DLL that implements the radio-management user-interface API and communicates privately with the NFP device driver. When the user changes the radio on/off state of the NFP device in the Windows Settings application, the NFP device driver receives notification of the change through the radio manager. The driver developer must define a private interface to communicate radio management state between the radio manager COM object and the NFP device driver. When the NFP device driver receives a command from the radio manager object to set the device's radio on/off state to off, the driver must save device's hardware state, turn off any RF components associated with the NFP device, and transition the device to D3. Note that the quantity and state of all handles from the GUID_DEVINTERFACE_NFP interface that are already open when the device enters D3 must be preserved while the device is in D3.

After the radio manager has indicated that the RF components in the NFP device should be enabled and powered on, the NFP device must immediately transition back to D0 when either of the following occurs:

- A client opens a new handle to the NFP device driver.
- Any already open handle receives the **IOCTL_NFP_ENABLE** notification.

The **IOCTL_NFP_ENABLE** notification will be sent by Windows to the NFP device driver when the platform exits Modern Standby and the user enters their password.

If the radio manager has indicated that the RF components in the NFP device should be disabled and powered off, the NFP device must remain in D3—even if a new handle is opened or an existing handle is enabled—until the radio manager indicates that the RF components should be enabled and powered on.

If the NFP device requires a command to be sent to return the NFP device to idle mode, this command can be sent over the communications bus (typically I2C) and should occur as part of the completion of the transition to D0 in the NFP device driver.

Transitioning the NFP device to D0 allows the _PS0 control method under the device in the ACPI namespace to be executed. If required, this control method can toggle a GPIO line from the SoC to the NFP device to return the device to idle mode.

Supported hardware power configurations

An NFP device should consume no more than one milliwatt of power, on average, while the hardware platform is in Modern Standby. If the device consumes no more than one milliwatt of power when it is in standby mode, the NFP device driver can place the device in standby mode when the device is not being used. Otherwise, the system designer must attach the device to a dedicated power rail that can be switched off when the device is not being used.

Standby power is less than one milliwatt

If the NFP device consumes less than one milliwatt of power in standby mode, the NFP device can be placed on any system power rail and can share this rail with any number of other devices. The device's transition to standby power mode occurs as part of its transition to D3.

The NFP device's entry to standby mode can be initiated either by sending a command over the communications bus (typically I2C) or by toggling a SoC GPIO line that is routed to the NFP device.

If the NFP device requires toggling of a GPIO line from the SoC to enter standby mode, the GPIO line must be described in a GPIO operation region in the ACPI namespace. The system integrator must implement the _PS3 and _PS0 methods under the NFP device in the ACPI namespace. The implementation of the _PS3 method should toggle the GPIO line to switch the NFP device to standby mode. The implementation of the _PS0 method should toggle the GPIO line to switch the NFP device from standby mode to idle mode. The toggling of this line might be subject to platform-specific or device-specific timing constraints.

As previously described, the NFP device driver must initiate a transition to D3 when all handles are closed, or all handles have received the **IOCTL_NFP_DISABLE** notification, or the radio manager has requested that the RF components in the NFP device be turned off.

Standby power is more than one milliwatt

If the power consumption of the NFP device in standby mode is more than one milliwatt, the NFP device must be placed on a dedicated power rail that can be independently switched on and off under control of a GPIO from the SoC. When all handles are closed, or all handles have received the **IOCTL_NFP_DISABLE** notification, or the radio manager has requested that the RF components be turned off, the NFP device

first transitions to D3, and then transitions to power-removed mode when the power rail is switched off.

The switchable power rail must be controlled by a GPIO line from the SoC or core silicon. The GPIO line should be described as part of a GPIO operation region in the ACPI namespace. In addition, the ACPI namespace must include a Power Resource that describes the switchable power rail and that implements _ON and _OFF control methods. The _ON and _OFF control methods toggle the GPIO line described in the GPIO operation region to turn the power rail on and off. _PR3 and _PR0 packages must be placed under the NFP device in the ACPI namespace and must link to the Power Resource that describes the power rail.

As previously described, the NFP device driver must enable D3cold. In addition, the driver must initiate a transition to D3 when all handles are closed, or all handles have received the **IOCTL_NFP_DISABLE** notification, or the radio manager requests that RF components be turned off. When the ACPI driver receives the D3 IRP, it will execute the _OFF method on the Power Resource indicated by the _PR3 package. This method will toggle the GPIO line that controls the power switching hardware to remove power from the NFP device.

If the radio manager has indicated that the RF components should be enabled, the NFP device driver must perform a transition to D0 when a new handle is opened or an existing handle has received the **IOCTL_NFP_ENABLE** notification. When the ACPI driver receives the D0 IRP (an **IRP_MN_SET_POWER** request of type **DevicePowerState**), it will execute the _ON method on the Power Resource indicated by the _PR0 package. This method will toggle the GPIO line that controls the power switching hardware to re-apply power to the NFP device.

Wake concerns

There are no wake concerns for the NFP device. The NFP device is not permitted to support waking the SoC while the platform is in Modern Standby.

Testing and validation

It is critical for the system designer to verify that the NFP device reliably enters standby mode or power-removed mode when the display is powered off for Modern Standby. The power management solution for the NFP device is highly dependent on the implementation of the third-party driver. Therefore, system integrators are encouraged to discuss the best method to test and validate the functionality of the NFP device with the NFP device vendor.

If the NFP device driver initiates a transition to D3 when all handles to the driver are closed or disabled, the XPerf tool can be used to observe this transition. This validation method is similar to that described elsewhere for touch controllers and sensor devices.

If the driver sends commands over the communications bus to transition the device into and out of standby mode, the driver should support event tracing so that the system integrator can easily validate power management operations. Instrumentation can easily be added to Windows drivers using Event Tracing for Windows (ETW) events. For more information, see Adding Event Tracing to Kernel-Mode Drivers. An advantage of ETW tracing is that that it can be viewed using the Windows Performance Toolkit suite of tools, which includes XPerf.

NFP device power management checklist

System integrators and NFP vendors should review the checklist below to verify that their system power management design is compatible with Windows 8 and above.

- Select an NFP device that has a standby mode that consumes an average of less than one milliwatt of power.
- Verify that the NFP device driver initiates a transition to D3 when all handles to the device are closed, or all open handles have received the IOCTL_NFP_DISABLE notification, or the radio manager requests that the RF components in the NFP device be disabled and powered off.
- Verify that the NFP device driver initiates a transition to D0 if the radio manager has indicated that the RF components should be enabled and subsequently the first handle is opened to the device or the first open handle receives the IOCTL_NFP_ENABLE notification.
- If the NFP device driver sends a custom command over the communications bus to tell the NFP device to enter standby mode, work with the NFP device vendor to determine the best way to verify that the NFP device enters standby power mode in response to this command.
- If the power consumption for the NFP device is greater than one milliwatt, ensure that the device is placed on a switchable power rail.
- If the NFP device is located on a switchable power rail:
  - Verify that the device supports completely removing power from the device when it's unused, and that restoring power automatically re-enables and re-initializes the device.
  - Verify that the power rail is controlled by a GPIO line from the SoC or core silicon.
  - Map the GPIO line into a GPIO operation region implemented in the system ACPI firmware.

    - Provide a Power Resource to describe the power rail, and include the _ON, _OFF, and _STA control methods in this Power Resource.
    - Provide the _PR0 and _PR3 packages under the NFP device in the system ACPI firmware and ensure that they reference the Power Resource.
    - If the NFP device requires a power-on delay before the driver can access it, encode this delay in the implementation of the _ON control method.
- Initiate multiple transitions into and out of Modern Standby, and then stress-test the operation of the NFP device when the display is turned on.

PCI Modern Standby Considerations

Article • 03/22/2021

This article will cover considerations for designers as they incorporate PCI into a Modern Standby system.

To meet DRIPS requirements, it is generally recommended that drivers of PCI devices opt in to D3Cold support whenever possible. For devices connected externally, like via Thunderbolt™, it is recommended that the driver support D3Cold in order for the system to enter DRIPS.

Below is an overview of generic PCI Power Management and information about D3Cold support. For D3Cold considerations specific to device classes, please refer to the individual device page under Device Specific Design.

Overview of PCI Power Management

PCI power management (PCI-PM) requires a collaboration between the bus driver and the device driver. In PCI-PM, the PCI bus driver is responsible for reading the PCI-PM registers and determines the capabilities of the device. It will also modify the PCI-PM registers to set the device into Dx states in response to the corresponding POWER IRPs. The device driver is responsible for requesting the power state change of the device as the Power Policy Owner. It must also save and restore the device specific context across power state changes.

Some on-SoC devices will also require the ACPI driver to participate in the power management flow, as they need ACPI Source Language Assembler (ASL) to completely power manage the device.

For more information about how PCI Power Management (PCI-PM) compliant hardware interacts with device drivers in the operating system and about how PCI-PM integrates with ACPI, please refer to PCI Power Management and Device Drivers.

Supporting D3cold in a Driver

It is recommended that root ports support D3 (End Point for D3.cold with L2/L3 ready). It is also generally recommended that device drivers opt in to D3Cold support whenever possible. On a Modern Standby system, a device that cannot idle to D3Cold might prevent other devices from entering low-power Dx states and thus prevent the system from entering DRIPS.

The device driver is the Power Policy Owner of the device and is responsible for determining whether its device can support D3Cold. Before enabling D3Cold for the device, the device driver must ensure its device can still operate correctly after entering D3Cold. For example, if the device requires wake capability while in D3Cold, the driver must verify that the system has sufficient support for that. For more information about how to determine the D3Cold capabilities of a device, please refer to D3Cold Capabilities of a Device.

The driver should use the following checklist to validate the platform support and make a decision whether to enable D3cold support for its device:

1. If the device does not require wake from D3Cold state, it is the driver's responsibility to determine whether the device supports D3Cold.
2. If the device requires wake from D3Cold state, a WDM driver should call GetIdleWakeInfo routine to determine the lowest-powered device power state from which the device can signal a wake event and decide whether to enable D3Cold accordingly. A KMDF driver should call the WdfDeviceAssignS0IdleSettings routine to enable the device to idle in the lowest-powered device power state from which the device can signal a wake event.

For more information about the parent bus driver and ACPI firmware's requirement to support wake from Dx states, please refer to the following document: Firmware Requirements for D3cold. For more detailed information about supporting D3Cold in a device driver, please refer to Supporting D3Cold in a Driver.

Sensors Power Management

Article • 05/03/2021

A mobile PC typically incorporates sensor devices such as an ambient light sensor (ALS), 3-D accelerometer, 3-D gyrometer, or 3-D magnetometer. When a sensor device is not being used by the operating system or by an application, power-management software can switch the device to a low-power mode to reduce power consumption. In a PC that supports the modern standby power model, sensor devices are expected to switch to a low-power mode shortly after the PC enters modern standby and remain in this mode until the PC exits modern standby.

This article explains how to implement power management for sensor devices. Additionally, this article discusses power management of the optional sensor microcontroller (also called the sensor fusion hub or the sensor MCU) and aggregated sensor devices. (e.g. A compass sensor device can be implemented by aggregating an accelerometer, a gyrometer, and a magnetometer under the control of a sensor microcontroller. The microcontroller exposes these sensor devices to Windows as a single logical sensor device.)

Sensors and sensor microcontroller

Sensor hardware is critical to the modern mobile experience. Starting with Windows 10, a rich system infrastructure is available to expose and manage multiple sensor devices. This infrastructure simplifies the development of applications that incorporate sensor information and that support critical built-in Windows scenarios, such as screen auto-rotation or changing the display brightness based on ambient light.

During system runtime, individual sensors can be powered off when they're not in use. The requirements for using a particular sensor device are communicated to the device and its drivers through the Windows Sensor API. When a sensor device is not being used by the operating system or by any applications, the device can be powered down by the sensor driver or by the firmware that is running on the sensor microcontroller.

After the system display turns off and the hardware platform enters modern standby, all sensor devices and optional sensor microcontrollers that are not already in low-power states should enter their low-power, standby states within a few seconds so that the platform as a whole can enter a low-power state. However, sensor drivers do not directly monitor transitions to and from modern standby to determine when sensor devices should be powered on and off. Instead, the sensor driver should enable the device to receive power when the device is actively being used by one or more clients, which can

be applications or operating system components. The driver should remove power from the device when no clients are using the device.

When the sensor class extension requests the driver to start reporting sensor sample readings, it calls the sensor driver's EvtSensorStart callback method. When the sensor class extension requests the driver to stop reporting sensor sample readings, it calls the driver's EvtSensorStop callback method. For more information, see About Sensor Driver Events.

After the computer enters modern standby and all sensor devices enter low-power states, the total power consumption of all system sensor hardware must be less than one milliwatt. The sensor devices and optional sensor microcontroller might enter a low-power standby state that is specific to the sensor hardware. Or, the hardware power rail to the sensor devices and the optional sensor microcontroller can be switched off under control of the sensor drivers and/or the system ACPI firmware.

Starting with Windows 10, support is provided for a limited set of sensor hardware connectivity options to the core silicon or System on a Chip (SoC) in a modern standby platform. The following sections detail the supported hardware and software configurations as well as their power-management behaviors both during modern standby and when the platform is actively being used.

Power management modes

Windows expects each sensor device or the sensor microcontroller to have three device power modes—active, idle, and standby—in addition to an optional, zero-watt, power-removed mode. The following table describes the power modes for a sensor device and optional sensor microcontroller. The table distinguishes between an idle mode in which the sensor hardware is being used but is currently idle, and an idle mode in which the sensor hardware is not being used.

Mode	Description	Average power consumption	Exit latency to active	Transition mechanism
Active	The sensor device and/or sensor microcontroller is actively providing or processing environmental changes.	< 100 milliwatts	N/A	N/A

Mode	Description	Average power consumption	Exit latency to active	Transition mechanism
Idle (in use)	The sensor device and/or sensor microcontroller is being used by one or more applications and is waiting to provide the next set of sensor information to the main processor.	< 50 milliwatts	Sensor-specific	Hardware autonomous
Idle (not in use)	The sensor device and/or sensor microcontroller is not being used by any application. Calibration data for the sensor or the sensor microcontroller is maintained.	< 5 milliwatts	Sensor-specific	Human Interface Device (HID) commands or Sensor Framework messages describing current usage of sensor devices.

Mode	Description	Average power consumption	Exit latency to active	Transition mechanism
Standby	The sensor device and/or sensor microcontroller is not being used by any application. Calibration data for the sensor or the sensor microcontroller is maintained. The sensor and/or sensor microcontroller does not take any further action until requested by software running on the main processor.	< 1 milliwatt (for all system sensors)	< 10 milliseconds	Multiple options: • HIDI2C command SET_POWER(Sleep) • Third-party driver private message • GPIO line from SoC to sensor hardware
Power-removed	Power is removed from the sensor device and/or sensor microcontroller and all hardware context is lost.	0 milliwatts	< 100 milliseconds	External entity removes power or applies power via ACPI firmware in response to D3 power IRP.

> **Note**
>
> In the preceding table, the term standby refers to a device power mode that is distinct from modern standby, which is a platform-wide power state.

Software power-management mechanisms

Run-time power management for sensor devices and the sensor microcontroller is predominately driven by whether they are being used. As a general rule, the sensor driver and hardware are expected to put a sensor in the idle power mode when it is not

being used by the operating system or an application. The Windows Sensor Platform provides information about the number of application or operating system clients connected to a given sensor, as well as requirements for the sensor's duty cycle or data rate. The sensor driver and/or hardware uses this information to seamlessly transition the sensor device to the idle power mode during times when the system is running and the display is turned on.

After the system display is powered off and the platform enters modern standby, Windows expects all sensors and sensor microcontrollers to enter a standby or power-removed mode.

The choice of a software power-management mechanism to use for sensor devices and the optional sensor microcontroller depends how the sensor hardware is exposed to Windows by the device driver, and how the sensor hardware is physically connected to the SoC or core silicon. Windows supports two methods of exposing and connecting sensor devices. One method uses the built-in sensor HID class driver over an I2C connection, where built-in HIDI2C driver transfers HID information over the I2C connection. The other requires a third-party driver that implements the Universal sensor driver interface and calls the methods in the SensorscxFunctions table.

The two options for connecting to a sensor or sensor microcontroller are compared in the following table. The selection of one of the two options for connecting to sensor hardware dictates the software power-management mechanisms that are needed to transition the sensor hardware to the standby or power-removed mode.

Connection option	Bus connection	Sensor driver required	Driver provider	Comments
HIDI2C	Sensor hardware connects directly to the SoC or core silicon over I2C.	Sensor HID class driver + HID-over-I2C class driver	Microsoft. Inbox component starting with Windows 8.	Pros / Cons
Third-party sensor driver	Sensor hardware connects directly to the SoC or core silicon over I2C or UART.	Third-party driver that implements **SENSOR_CONTROLLER_CONFIG**	Sensor device vendor.	Pros / Cons

HIDI2C

For the HIDI2C option, the optional sensor microcontroller is physically connected to the SoC or core silicon through an I2C bus. The microcontroller exposes multiple top-level HID collections, one for each logical sensor device. For example, a compass sensor can be exposed through HID as a logical sensor device that is an aggregation of the accelerometer, gyrometer, and magnetometer sensors behind the sensor microcontroller. This is the easiest to implement in terms of connectivity and software as it does not require third-party software for the sensor device.

The Windows HIDI2C stack is similar to that of touch controllers and pen digitizers in that it supports two software power-management mechanisms: an in-band HID command and a run-time transition to the D3 state.

In-band HID command

SET_POWER(Sleep) Sent to the device after the display turns off and the platform enters modern standby. This command can transition the device to the standby power mode.

SET_POWER(On) Sent to the device when the platform exist modern standby and the display is turned on again.

Run-time transition to D3 state for the HID sensor device stack

D3 IRP A IRP_MJ_POWER request that is sent to the driver stack for the device immediately after the SET_POWER(Sleep) command. This tells the device to enter the D3 device power state. As part of the transition to D3, the system ACPI firmware might run control methods to switch the device into the standby or power-removed mode.

D0 IRP A IRP_MJ_POWER request that is sent to the driver stack for the device when the platform exists modern standby and the display is turned on. This tells the device to enter the D0 device power state. If necessary, the system ACPI firmware can run control methods to switch the device back to the idle (not-in-use) mode.

Third-party sensor driver

For the third-party sensor driver, the sensor microcontroller is physically connected to the core silicon through an I2C bus or UART.

The sensor device vendor must provide a User-Mode Driver Framework (UMDF) driver that implements **SENSOR_CONTROLLER_CONFIG** interface. The UMDF driver communicates with the sensor device over I2C or UART. This can be implemented

multiple times—one time for each sensor that is behind the sensor microcontroller. The third-party sensor driver is responsible for creating and coordinating all power management.

Third-party sensor drivers are expected to be built by using the Windows Driver Frameworks (WDF) and be based on the Adxl354acc sample driver. The driver must use a power-managed queue and configure the D3 idle state through a call to the **IWDFDevice3::AssignS0IdleSettingsEx** method. The driver should use the **IWDFDevice2::StopIdle** and **IWDFDevice2::ResumeIdle** methods to indicate to WDF when the device is idle or active. The driver should also enable D3cold by setting the **ExcludeD3Cold** member of the **WDF_DEVICE_POWER_POLICY_IDLE_SETTINGS** structure to **WdfFalse**. Enabling D3cold allows the platform to remove power from the sensor device after it becomes idle and enters the D3 state.

As a best practice, put device-specific code in the driver and put platform-specific code in the ACPI firmware to enable low-cost re-use of driver code across multiple platforms.

Entering modern standby requirements

The requirements of the third-party sensor driver for power management are a function of the standby power consumption of the sensor hardware.

Third-party sensor drivers should initiate a transition to D3 when the sensor device is ready to enter the standby or power-removed mode, even if the device is capable of using an in-band communications mechanism to switch to a power mode that consumes less than a milliwatt. The reason for this requirement is that many bus drivers in Windows track the device power state of their endpoint devices and power down only when all endpoint devices have powered down. For some SoC designs and connection buses (notably Universal Serial Bus (USB)), all endpoint devices and the host controller must be in D3 for the SoC to enter the lowest power state during modern standby. Inability to enter the lowest power state can easily prevent a system from meeting modern standby requirements for battery life.

If the sensor hardware has a standby power consumption of less than one milliwatt for all controlled sensor hardware, the sensor driver should automatically switch the device to the standby mode when the sensors (or all sensors on the microcontroller) are no longer in use.

If the sensor hardware has a standby power consumption of greater than one milliwatt, the sensor driver must perform a D3 transition and allow ACPI control methods to remove power from the sensor device. The sensor driver must save all required sensor device state so that power can be removed from the device during D3. The sensor

hardware vendor should collaborate closely with the system integrator to ensure that the sensor hardware and driver performs the D3 transition reliably and quickly.

> **Important**
>
> The driver must save all sensor device context before the device enters D3, and must restore all sensor device context after the device enters D0.

Shortly after entry to modern standby, Windows automatically stops the use of all sensors by disabling the operating system's use of sensors (for example, ambient light and rotation) and by suspending applications. The sensor driver should aggregate the state of all controlled sensor hardware and switch this hardware to the standby device power mode when all sensors are no longer in use.

The mechanism to switch the sensor device to the standby mode can be designed to use in-band communication through the bus that connects the device to the SoC. For example, a proprietary standby command can be sent over the bus to the sensor hardware. Or, the sensor hardware might be connected to a GPIO line that switches the device into and out of standby mode.

> **Note**
>
> When a GPIO line is used to switch the device to standby mode, the sensor driver should transition the driver stack to D3 and allow the ACPI control methods for the device (for example, _PS3) to set the GPIO line to the state required to place the hardware in standby mode. This scheme allows the sensor driver to be written in a platform-agnostic manner—the specific GPIO line, timing requirements, and other platform-specific information are encoded in the ACPI firmware supplied by the system integrator and not in the device-specific driver.

Exiting modern standby requirements

When the platform exits modern standby, the sensor driver must transition the sensor hardware back to the idle (not-in-use) mode. As system services resume, Windows will request the use of sensors, such as rotation and ambient light, that are needed to perform system functions. As applications resume, they might request sensor information. If the sensor hardware requires an in-band message to return the device to the idle mode, the device driver should send this message as soon as the first request for sensor information is sent. If the sensor hardware requires a GPIO line to signal the device to return to the idle state, the driver must use this GPIO line to perform a

transition to D0 as soon as the first request for sensor information is provided. In this case, ACPI control methods (for example, _PS0) should toggle the GPIO line as required to initiate the transition. Finally, if the sensor hardware previously required a transition to the power-removed mode because the power consumption in standby mode exceeds one milliwatt, the sensor driver must perform a transition to D0 and allow ACPI control methods to restore power to the device.

Supported hardware power configurations

The hardware power-management configuration to use for a sensor device depends on the sensor hardware's power consumption in standby mode and whether an optional sensor microcontroller manages the device.

Standby power < one milliwatt

If the power consumption of a sensor device in the standby power mode does not exceed one milliwatt, the platform designer is not required to attach the sensor hardware to a power rail that can be turned on and off by ACPI control methods. One of the following mechanisms is used to switch the sensor to the standby power mode:

- A HID SET_POWER(Sleep) command.
- A GPIO line from the SoC.
- A proprietary command sent to the sensor hardware by the third-party sensor driver.

If the platform includes a sensor microcontroller, the microcontroller chip either might contain one or more integrated sensor devices or might be connected to one or more external sensor devices. In either case, these sensor devices are, from the software point of view, hidden behind the microcontroller and invisible to Windows. If a sensor microcontroller and its aggregated sensor devices together consume less than one milliwatt when the microcontroller and sensor hardware are in the standby power mode, the platform designer is not required to attach the microcontroller or sensor hardware to a power rail that can be turned on and off by ACPI control methods. The sensor microcontroller uses one of the following mechanisms to transition itself and all the sensors that it manages to and from standby mode:

- An HIDI2C SET_POWER (or similar) command sent over the communications bus.
- A GPIO line from the SoC.

If the sensor requires a GPIO line from the SoC to initiate transitions to and from standby mode, the platform firmware must provide both a _PS3 object and a _PS0 object in the ACPI namespace under the sensor hardware device. The ACPI firmware

must also include a GPIO operation region that describes the GPIO line from the SoC to the sensor hardware. The _PS3 control method toggles the GPIO line to switch the device to the standby mode, and the _PS0 control method toggles the GPIO line to switch the sensor hardware to the idle mode.

The following block diagram shows the power-management options for a standalone sensor that consumes less than one milliwatt in standby power mode.

One option is to use the Windows HIDI2C stack, as shown on the left side of the preceding diagram. In this case, the sensor's transition to standby power mode can be initiated either by an in-band HID SET_POWER(Sleep) command or by a D3 IRP that the ACPI driver handles by executing the _PS3 control method for the sensor.

The other option is to use a third-party sensor driver, as shown on the right side of the preceding diagram. The third-party sensor driver can initiate the transition to standby power mode either by using an proprietary in-band command or by sending a D3 IRP that the ACPI driver handles by executing the _PS3 control method for the sensor.

The platform designer can choose either mechanism regardless of whether the sensor devices are integrated or external to the microcontroller chip.

Standby power > one milliwatt

If the platform includes sensor hardware and/or a sensor microcontroller that together consume more than one milliwatt in standby power mode, the sensor hardware and microcontroller must be transitioned to the power-removed mode when the system is in modern standby. In this configuration, the sensor, the optional sensor microcontroller, and any sensors behind the microcontroller should be placed on a single power rail that is switched on and off under the control of a GPIO line from the SoC.

This configuration requires the platform designer to place all sensor hardware on a switchable power rail, controlled by a GPIO line from the SoC. If multiple input voltages are required for the sensor hardware, multiple switches, each controlled by the same GPIO line, can be used. In addition to the switchable power rail, the platform ACPI firmware must define a Power Resource in the namespace. This Power Resource describes the sensor hardware, and includes the _ON and _OFF methods that are responsible for using a GPIO operation region to toggle the GPIO line from the SoC.

The platform firmware must include a reference to the Power Resource under each sensor device in the ACPI namespace on the switchable power rail, including _PR0 and _PR3 objects.

The following block diagram shows the power-management options for sensor hardware and/or a sensor microcontroller that together consume more than one milliwatt in standby power mode. The two options are to use the Windows HIDI2C stack, as shown on the left side of the diagram, or to use a third-party sensor driver, as shown on the right side.

In the configuration that uses the built-in HIDI2C driver stack, as shown on the left side of the preceding diagram, the HIDI2C driver initiates a D3 transition after the display turns off and the platform enters modern standby. When the D3 IRP flows through the ACPI driver, the _PR3 object will be evaluated and Windows will turn off the specified Power Resource by running the _OFF method. If multiple sensors share the Power Resource, Windows automatically reference-counts all of the sensors and runs the _OFF method only after all sensors have entered D3.

If the sensor hardware uses a third-party sensor driver, as shown on the right side of the preceding diagram, the control flow is the same as before, except that the sensor driver is responsible for initiating the transition to D3.

After the platform resumes from modern standby and an application or the operating system requests use of the sensor, the driver transitions to D0. A D0 IRP flows through the ACPI driver and the _PR0 object is evaluated so that the ACPI driver executes the _ON method for the associated Power Resource. The _ON method toggles the GPIO line to turn on the switchable power rail. If the system uses a third-party sensor driver, the driver must request a D0 IRP and initiate a transition to D0 immediately after sensor data is requested by the operating system or an application.

Wake concerns

There are no wake concerns for sensors or the optional sensor microcontroller. Sensor devices are expected to be in the standby or power-removed mode during modern standby and are not expected to wake the SoC while the platform is in modern standby.

Testing and validation

It is critical for the system designer to verify that the sensor hardware enters the standby or power-removed mode when the display is powered off for modern standby. The method used to test and validate device power management depends on how the sensor device is connected.

HIDI2C-connected sensor

If the system uses the Windows HIDI2C stack, the system integrator should contact the sensor driver vendor for information about how to best verify that the driver correctly performs power management. Sensor driver vendors are encouraged to use Event Tracing for Windows (ETW) tracing for all power-management decisions in their device driver, and to provide sample documentation to system integrators to describe how to verify correct power-management operation using the ETW events and the Windows Performance Toolkit (WPT).

Third-party sensor driver

If the system uses a third-party sensor driver, the system integrator should contact the sensor driver vendor for information about how to best verify that the driver correctly performs power management. Sensor driver vendors are encouraged to use Event Tracing for Windows (ETW) tracing for all power-management decisions in their device driver, and to provide sample documentation to system integrators to describe how to verify correct power-management operation using the ETW events and the Windows Performance Toolkit (WPT).

If the driver initiates a transition to D3 when all of its sensor devices are no longer being used, you can follow the instructions in the following list to verify that this transition occurs as expected, and that a sensor device returns to D0 when an application or the operating system needs to use the device again.

The software-focused method uses Windows instrumentation to verify that the D3 IRP passes through the device driver stack for the sensor device. The Windows power manager has built-in ETW instrumentation, which includes instrumentation for detecting Dx IRPs (device power requests). To view this information in a manual mode, download the Windows Performance Toolkit and install it on the system under test.

After installing the Windows Performance Toolkit, follow these instructions to start a user-mode XPerf trace:

1. Open a Command Prompt window as Administrator.
2. Browse to the **\%ProgramFiles%\Windows Kits\8.0\Windows Performance Toolkit** folder.
3. To start Xperf, run the following command: `xperf.exe -start power_session -on Microsoft-Windows-Kernel-Power`
4. Transition the system to modern standby by pressing the power button.
5. Wait 30 seconds.

6. Transition the system out of modern standby by pressing the power button.
7. Run the following command to stop event logging: `xperf.exe -stop power_session`
8. Convert the binary trace file to .csv and human-readable format: `xperf.exe -i \user.etl > power.txt`
9. Open the Power.txt file in a text editor and search for the hardware ID of the sensor device. You can look up the hardware ID of the sensor device in the **Details** tab of the device properties in Device Manager under **Device Instance Path**. In the following example the device instance path of the sensor device is ACPI\MST0731\2&daba3ff&0.
10. The initiation of the D3 IRP for the sensor device is indicated by an event of type Microsoft-Windows-Kernel-Power/IRP/Stop that has the device instance path of the sensor device and a last event value of 3, which indicates that the target state is D3. The following output event from the Power.txt file shows the start of the D3 IRP. The last two parameter values for this event (shown at the far right) indicate the device instance path and target state.

    ```
    Microsoft-Windows-Kernel-Power/Irp/Start, 7605393, "Unknown" (4), 256, 0,,,,, 0x868e2728, 1, 2, 0x85fb56e0, 25, "ACPI\MSFT0731\2&daba3ff&0", 3
    ```

11. This event should be logged near the start of the Power.txt output file. The parameter value `0x868e2728` in the preceding output event is a pointer to the IRP structure for the D3 IRP. By searching for subsequent events in the trace file that have this same IRP pointer, you can follow the progress of the D3 IRP as it flows through the driver stack for the sensor device.
12. `Microsoft-Windows-Kernel-Power/Irp/Start, 7605393, "Unknown" (4),256, 0,,,,, 0x868e2728, 1, 2, 0x85fb56e0, 25, "ACPI\ATML1000\2&daba3ff&0", 3`
13. `Microsoft-Windows-Kernel-Power/Driver/Start, 7605416, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x85fb56e0, "\Driver\sensdrv"`
14. `Microsoft-Windows-Kernel-Power/Driver/Stop, 7605515, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x85fb56e0`
15. `Microsoft-Windows-Kernel-Power/Driver/Start, 7605522, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x85fab6a0, "\Driver\i2cdrv"`
16. `Microsoft-Windows-Kernel-Power/Driver/Stop, 7608342, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x85fab6a0`

17. `Microsoft-Windows-Kernel-Power/Driver/Start, 7608351, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x857ffb90, "\Driver\ACPI"`
18. `Microsoft-Windows-Kernel-Power/Driver/Stop, 7608416, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x857ffb90`
19. `Microsoft-Windows-Kernel-Power/Driver/Start, 7608424, "Unknown" (4), 20, 0,,,,, 0x868e2728, 0x85fb56e0, "\Driver\sensdrv"`

When the Windows ACPI driver, Acpi.sys, processes the D3 IRP, Acpi.sys runs the corresponding _PR3 control method. The system firmware designer provides this control method to indicate which Power Resource must be turned off for the sensor device to enter the D3 state. Acpi.sys also executes the _OFF control method under the Power Resource.

You can use a similar process to verify that the sensor device returns to D0 when the platform exits modern standby and the display turns on. A Microsoft-Windows-Kernel-Power/IRP/Start event for the sensor device will be logged with a target state of 0 (indicating D0) immediately after the power button is pressed to wake the system, and the operating system or a resumed application requests sensor data.

Sensor and sensor microcontroller power-management checklist

System integrators and sensor device vendors should use the following checklist to ensure that their system power-management design is compatible with Windows 8 and higher.

- Select sensor hardware that is compatible with the built-in HIDI2C driver and HIDSensor driver stack.
- Select sensor hardware that has a standby power consumption of less than one milliwatt.
- Verify that the sensor hardware and third-party driver (if required) support run-time idle power management when the display is turned on:
  - Sensors should be powered down and enter D3 automatically when they are not being used by an application or the operating system.
  - Sensors should power on and enter D0 automatically when sensor data is requested by an application or the operating system.
  - The third-party sensor driver should be implemented as a WDF driver and can be based on the SpbAccelerometer sample driver.

    - Polling for sensor information should be limited and enabled at the lowest power consumption level possible. For example, polling an analog sensor should happen behind a microcontroller or other low-power control hardware, which can interrupt the SoC when new sensor data exceeds some threshold detection value. Avoid polling the sensor in a driver that periodically runs on the SoC, which can significantly increase overall system power consumption.
- If the sensor hardware uses a third-party driver:
    - The system integrator should communicate with the sensor device vendor to understand how to implement power management for the sensor hardware.
    - If the sensor hardware consumes more than one milliwatt in standby power mode, place the sensor hardware on a standalone power rail that is controlled by a GPIO line from the SoC. Provide references to the required ACPI Power Resource, _ON/_OFF control methods, and Power Resource under the sensor device in the ACPI namespace (as described below).
    - If the sensor hardware uses a GPIO line from the SoC to switch the device to standby power mode, make sure that the system ACPI firmware includes the proper _PS3 and _PS0 control methods (as described below).
- If the sensor hardware includes a sensor microcontroller that has sensor devices connected behind it, the sensor microcontroller must have a way to power down the sensor devices. The devices can be powered down by using either in-band communication over the bus that connects the microcontroller to the devices, or a GPIO line from the microcontroller to the devices.
- If the sensor hardware requires a GPIO line from the SoC to switch the device to the standby power mode:
    - Make sure the GPIO line from the SoC meets the level and trigger requirements set by the sensor hardware vendor.
    - In the ACPI namespace, describe the SoC GPIO pin as part of a GPIO operation region.
    - Provide a _PS3 control method under the sensor device in the ACPI namespace to toggle the signal on the GPIO line as needed to switch the sensor hardware to the standby power mode.
    - Provide a _PS0 control method under the sensor device in the ACPI namespace to toggle the signal on the GPIO line as needed to switch the sensor hardware to the idle or active mode after the device switches to D0.
- If the sensor hardware consumes more than one milliwatt in the standby power mode:
    - Place all sensor hardware on a power rail that can be switched on and off by a GPIO line from the SoC. Or, if the platform contains multiple sensors that have different supply voltage requirements, provide separate rails that can be independently switched.

    - Describe the switchable power rail as a Power Resource in the ACPI namespace.
    - Under this Power Resource, provide the _ON and _OFF control methods that toggle the power rail on and off by using a GPIO line that is described as part of a GPIO operation region.
    - In the ACPI namespace, provide _PR3 and _PR0 objects that designate the Power Resource for the sensor hardware.
    - Make sure that the _ON and _OFF methods incorporate any de-bounce or timing requirements of the sensor hardware.
- Test and validate the run-time power management of the sensor devices in the platform. Work closely with the sensor hardware vendor to validate run-time power management when the system display is on.
- Test and verify that the sensor hardware enters the standby or power-removed mode when the platform enters modern standby.
    - If the sensor hardware uses the HIDI2C + HID sensor driver stacks included with Windows, see Testing and validation for details.
    - If the sensor hardware uses a third-party driver, contact the sensor driver vendor for the recommended test methodology.
    - If the sensor driver performs a transition to D3 as part of its entry to the standby or power-removed mode, use the Windows Performance Toolkit as described in Testing and validation. Verify that the sensor hardware enters D3 when the platform enters modern standby, and that the sensor hardware enters D0 after the system exits modern standby and sensor information is requested again.
- Measure the power consumption of the sensor hardware in the standby or power-removed mode.
- Initiate multiple transitions into and out of modern standby, and then stress-test the operation of the sensor devices and the applications that use sensor information when the display is turned on.

Storage Design

Article • 11/05/2020

This section will describe considerations for designers as they incorporate storage device into a Modern Standby system.

Adaptive idle usage

The new adaptive idle timeout algorithm, introduced in Windows 10, balances power consumption with device reliability from a host-perspective. It applies only to storage devices reporting a seek penalty and will only be active if the respective miniport driver opts in to this capability. StorAHCI.sys (inbox driver) automatically enables this.

Since rotating media can only withstand a finite number of load/unload cycles – which are caused by power state transitions from D0 to D3 and back – the host will attempt to average no more than one power state transition per 5-minute interval, by default.

This means, if a device has been powered off (D3) for a long period of time (for example 30 minutes), then you could expect the next power state transition from D0 to D3 to be very fast, since the host built up "credit" and the average would still be well below one transition per 5 minutes.

On the other hand, if the device has been power cycled more frequently than once per 5 minutes, the host will delay the next transition from D0 to D3 by as much as 5 minutes in an attempt to return to the target average over time.

A miniport driver can opt-in to this behavior:

- Register a **STOR_POFX_DEVICE_V3** with **StorPortInitializePoFxPower** in the ScsiUnitPoFxPowerInfo unit control callback.
- Set the STOR_POFX_DEVICE_FLAG_ADAPTIVE_D3_IDLE_TIMEOUT flag in the **Flags** member of the **STOR_POFX_DEVICE_V3** structure.
- Set the **MinimumPowerCyclePeriodInMS** member in the **STOR_POFX_DEVICE_V3** structure to the value that indicates, in the worst case, how often the device can be power cycled. Units are in milliseconds. We recommend this be at least 5 minutes (i.e. the device should not be power cycled more than once every 5 minutes).

Storage reliability

In order to test for the impact of Modern Standby on rotational device reliability, you can following these general guidance:

1. Query the current count of Load/Unload Cycles on the rotational device.
   a. You can do this through PowerShell and the `Get-StorageReliabilityCounter` cmdlet, which has a field called: `LoadUnloadCycleCount`
   b. `Get-PhysicalDisk | Get-StorageReliabilityCounter | fl`

2. Execute a connected or disconnected Standby workload typical for your target customer or audience.

3. After the system exits Modern Standby, requery the Load/Unload cycle count.

By dividing the change in Load/Unload cycles by the time spent in Modern Standby, you can compute a ratio of Load/Unload cycles per time period. As an example, for a device spec'ed with a maximum of 600,000 Load/Unload cycles to reach a 5 year warranty, no more than 120,000 cycles can be incurred per year. This comes to a ratio of ~0.23 Load/Unload cycles per minute that the system should not exceed.

Note Different systems are targeted for many different audiences and usage patterns. Therefore, it is the OEM's responsibility to ensure the connected or disconnected Modern Standby system (they build) can handle the expected usage patterns.

Integrating apps with modern standby

Article • 05/18/2022

Most apps that are built for Windows do not need any additional support to run in a modern standby system. Microsoft Store apps are already designed to separate background activity from the interactive pieces of the app by using background tasks. Many Microsoft Store apps use background tasks to update their live tile, generate toast or notifications, and sync data with the cloud. These same background tasks can run during modern standby.

Desktop applications are automatically paused by the Desktop Activity Monitor (DAM) after the system enters standby. The DAM suppresses desktop application execution in much the same way as the Sleep (S3) system power state in systems that support the traditional ACPI S3 and S4 states. Any differences between modern standby and S3 are largely transparent to desktop applications. These applications require no modification or update to ensure that they keep running after the system exits standby.

The following sections describe the effect of modern standby on the execution of Microsoft Store apps, desktop applications, and system services.

Microsoft Store apps

Microsoft Store apps can create background tasks by using a specific set of WinRT APIs that allow the task to execute on a regular schedule or in response to an incoming network event. (WinRT is the Windows Runtime API surface and is distinct from Windows RT, which is the version of Windows that runs on the Arm processor architecture.) The WinRT APIs for background tasks allow the app's runtime to be carefully managed by Windows to prevent excessive battery drain.

A common use of background tasks is to update the content of tiles on the Start page in response to network events. In a modern standby system, the user sees the updated tile as soon as the system turns on; that's because the background task ran during modern standby.

Note that background tasks may be paused or throttled during Modern Standby on DC power unless the app has been set as always allowed to run in the background (in Settings > System > Battery > Battery usage per app > Choose when this app can run in the background).

Apps can also use timer-triggered background tasks to periodically check for new app-specific content during modern standby. Note that timer-triggered background tasks

will only run during Modern Standby on AC power; they will not run on DC power. The execution times of an app that uses a periodic timer during modern standby are automatically coalesced with the execution times of other apps. The expiration of the timer is limited to be no more frequent than once every 15 minutes, and the app has a very limited amount of CPU time available for the update.

For more information about background tasks, including how to create and register them, see Support your app with background tasks.

Desktop applications and system services

Desktop applications typically require no extra work to integrate with modern standby.

The Desktop Activity Moderator (DAM) is the Windows component that pauses all desktop applications and throttles the runtime of third-party system services during modern standby. The purpose of the DAM is to maintain basic software compatibility with existing applications and services, but mitigate their impact on battery life during sleep.

Windows prevents desktop applications from running during any part of modern standby after the DAM phase completes. Windows allows third-party system services to execute in a throttled mode after completing the DAM phase. However, as of 24H2, throttling may be stopped to decrease battery drain.

Additional resources

For more information about the effect of modern standby on app execution, see the following topics:

- Modern Standby
- Prepare software for modern standby
- **PowerSetRequest**
- **PowerClearRequest**

Power configuration options

Article • 05/19/2021

The configuration options described in this topic are available on all Modern Standby systems (both connected and disconnected). These options were previously unavailable on Windows 8.1 Connected Standby systems.

Note S3 Sleep and Hibernate are not available on Windows 10 Mobile.

Hibernate

The ability to trigger a system to doze to hibernate from Modern Standby is exposed in Windows 10. By default, this behavior is off, but can be set by users through the advanced control panel, similarly to setting this behavior on a legacy S3 system. Care should be taken when using this functionality. In particular, the following factors need to be considered:

- If the storage size on the system is extremely small, care should be taken to ensure the user experience is acceptable with hibernate enabled. Hibernate requires the creation of a hiberfile. In current builds, the default hiberfile size is 80% of the system memory size. We are investigating creating a default hiberfile size that is smaller for systems with a primary drive under 128 GB.
- Once the system dozes to hibernate, background activity, and “always on”-related functionality stops. The doze timer should be set high enough to ensure that a user would find it acceptable that they would not have such functionality.

Please note that on clean installs of the May 2019 update and later releases (build 18362+), Adaptive Hibernate is on by default. For more information on Adaptive Hibernate, please see this page.

S3 Object presence in ACPI firmware

From a user experience perspective, the S3 and Modern Standby experiences were mutually exclusive on the same hardware. When the user selects Sleep in the OS, only one or the other could happen.

It is expected that a system could be configured to have both an S3 object and a populated FADT table (ACPI S0 Low Power Idle flag, same for Windows 8.1 Connected Standby and Modern Standby), making it possible for enterprise users to change to Windows 7 without requiring any BIOS changes.

Display, sleep, and hibernate idle timers

Article • 09/25/2020

Traditional sleep (S3) and Modern Standby systems implement both a display idle time-out and a sleep idle time-out. The reason for providing two separate time-outs is to allow the system to stay turned on and fully running, but save power by turning off the display. If a user sets the two timeouts to be the same on a modern standby PC, the power model is similar to that used by most smartphones: when the screen is off, the system is always in a low-power mode and always connected to the Internet.

Mobile systems also support the ACPI Hibernate (S4) state. Hibernate is used to reduce battery drain during sleep by saving all information in memory to the disk and powering off the system completely. Although Modern Standby systems support Hibernate (S4) state, it is not entered automatically after a fixed amount of time in sleep. Instead, Windows manages Hibernate intelligently, only using it when required to preserve user's battery life. That's because modern standby is a consistently low-power state with long battery life. A modern standby PC is similar to a smartphone-- few users would tolerate their smartphone periodically turning off and entering hibernation. Instead users would expect the system to stay turned on and connected but operate in a low-power mode.

The following table summarizes the differences between a modern standby PC and a traditional sleep PC.

Timer	On a traditional PC	On a modern standby system	Notes
Dim the display	This time-out is no longer supported starting with Windows 8.1. In Windows 10, the display is automatically dimmed 10 seconds before it is turned off.	This time-out is no longer supported starting with Windows 8.1. In Windows 10, the display is automatically dimmed 10 seconds before it is turned off.	

Timer	On a traditional PC	On a modern standby system	Notes
Turn off the display	When this time-out occurs, the display immediately turns off. However, the system continues to run and all applications continue to operate normally as if the display was powered on.	When this time-out occurs, the display immediately turns off. However, the system continues to run and all applications continue to operate normally as if the display was powered on.	There is a 5-second grace period between when the display turns off and when the screen off notification is fired.
Put the computer to sleep	When this time-out occurs, the system immediately enters the ACPI Sleep (S3) state. All applications stop execution because all processors will be powered off.	When this time-out occurs, the system enters modern standby and the display is immediately powered down. Then the system steps through the preparation phases described in Prepare software for modern sleep.	
Hibernate after	When this timer expires, the system wakes from ACPI Sleep (S3) and immediately enters ACPI Hibernate (S4) to save power.	This time-out is not recommended for modern standby systems.	Modern standby has the intelligence to manage hibernate internally and setting a separate hibernate timeout can result in a loss of some features.

Advanced power settings

Article • 09/25/2020

Windows has built-in extensibility for power policy settings. The purpose of this support is to allow device vendors and software developers to extend the power policy model to their own hardware and applications.

In Windows 10, the configuration of these settings is exposed to the user in the **Power Options** control panel as the **Change plan settings** option. Many Windows power policy settings are also configurable through the **Change advanced power settings** option, including Wi-Fi power management policy and battery threshold and action policy.

In a modern standby system, the following power policies are displayed in the **Advanced settings** tab, and all other policy policies that exist on a system that uses the traditional ACPI Sleep (S3) and Hibernate (S4) states are removed:

- Desktop background slideshow
- Battery levels and actions
- Sleep, display, and hibernate timeouts

Windows OEM policy for power settings

Article • 11/05/2020

In previous versions of connected standby, the number and type of power policies that the system manufacturer can configure is greatly reduced, compared to a system that uses the traditional ACPI Sleep (S3) and Hibernate (S4) states.

However, Windows 10 introduces new and enhanced imaging and deployment tools that allow packaging, servicing, and imaging convergence through tooling convergence. A new Windows image customization framework that supports image customizations across all Windows 10 editions is now available to provide a unified image customization and configuration experience. For more information about this unified customization framework, see Customize using the Windows Provisioning framework.

To learn about how to configure the power settings using Windows provisioning, see Configure power settings. Each power setting topic includes the identification GUID, allowed values, meaning, and common usage scenarios for the setting.

Optimizing modern standby

Article • 09/25/2020

To optimize power savings during modern standby, start by reducing the amount of power that is consumed during the power floor—the state in which all components are idle and inactive and power is dominated by hardware static leakage. After the power floor is optimized, the power consumed by the Wi-Fi and communications devices can be reduced.

During modern standby, a well-behaved platform should spend most of its time operating at the power floor. The system designer must ensure that the Wi-Fi and communications devices do not unnecessarily wake the System on a Chip (SoC), which causes additional activity in the operating system and apps.

Optimizing power consumption

The power floor is the least amount of power that is required by the system during modern standby. The power floor is measured when the system is in airplane mode, and when:

- The SoC is in its lowest power state (deepest runtime idle platform state, or DRIPS).
- Memory is in self-refresh.
- Wi-Fi and communications devices are in their radio-off states.
- Devices outside of the SoC are in their low-power states (D3 or D3cold).

The power floor is a key metric to measure and optimize for modern standby power consumption. The system should spend the majority of its modern standby time—typically well over 90 percent of a modern standby session—in the power floor state. Every milliwatt that is eliminated from the power floor significantly improves modern standby battery life.

To measure the power floor on a modern standby system, you need the following setup:

- An instrumented system that can measure power consumption at several power rails, which include the following:
  - Total system power
  - SoC and DRAM
  - The Wi-Fi/Bluetooth device
  - The mobile broadband device (if equipped)
  - The sensor hub
  - The touch controller

    - Any other microcontroller (such as the keyboard, touchpad, custom battery controller, or legacy embedded controller)
- A power meter device that is capable of reading the power instrumentation.
- Software that is capable of reading from the power meter and tracking power consumption over time.

You can use your own instrumentation methodology to take power measurements. For the power rails, we recommend that you take power measurements at a sample rate of 1,000 hertz or greater with at least 1-milliwatt accuracy.

In addition, you might have a larger number of test systems that are designed to measure power consumption only at the system level. These test systems are best used for regression testing before updates to firmware and drivers are made to identical systems that were previously shipped and must be maintained in the field.

To measure the power floor, connect the power meter to the pin-out wires on the test system. Then use the software power-measurement tool to start capturing power readings. Most software power-measurement tools allow you to export all the power readings taken during the test to a .csv file.

Not all readings in the .csv file represent power floor values. These readings also include brief periods when the system is performing activities. To filter out active periods, consult with your SoC vendor to pick a reasonable number to use as a threshold for the power floor. Then filter out all readings that are above this number.

The readings that remain after filtering are a safe estimate of time periods when the system is in the lowest power state. Take the average of these readings to estimate the power floor for the scenario.

The following steps summarize the process to optimize the power floor for a system:

1. Consult with your SoC vendor to determine the expected power floor for the SoC + DRAM.

2. Consult with the hardware designer to determine the expected power floor for the entire system, including all components outside of the SoC and DC-to-DC conversion losses.

3. Perform power measurements and capture power readings to measure the power floor for your system.

4. Compare the power floor from step 3 with the expected power floor in steps 1 and 2.

5. If this comparison reveals any discrepancies, run a SleepStudy report and view the percentage of time that the system spends in software and hardware low-power states:
   a. If there are software components that prevent the system from entering the lowest power state, consult with the owners of these components to determine how to reduce their activities.
   b. If no software components are active but some hardware components are active, consult with the device driver vendor or Microsoft to identify the problem.
   c. If both software and hardware allow the system to stay in the lowest power state a majority of the time, look at the power that is consumed on each power rail to identify any components that consume more power than expected. After these components are identified, you might need to consult with each device hardware vendor to diagnose the problem.
6. Repeat steps 2 through 4 until the power floor is within the expected range from step 1.

The power floor is the baseline metric that is used to measure the power for key modern standby scenarios. It is used as a reference to gauge whether a particular modern standby scenario is consuming the expected power. To establish a baseline expectation, you'll need to get the power floor for modern standby working correctly in airplane mode before you progress to more advanced modern standby scenarios.

Any optimizations made to the power floor have a cascading effect on all scenarios that depend on it and are crucial in improving overall system power.

Optimizing modern standby Wi-Fi connectivity

Optimizing the power consumption of modern standby with Wi-Fi connected requires measurement and investigation of these two separate problem areas:

- Wi-Fi radio power consumption.
- Extra system activity due to a network connection.

Measuring and optimizing Wi-Fi radio power consumption is critical to modern standby because most modern standby-capable PCs are put in modern standby with the Wi-Fi radio connected. Moreover, the Wi-Fi radio is one of the few devices outside of the SoC that is capable of very high power consumption when the SoC is in its low-power (DRIPS) state.

The Wi-Fi radio can use several power-saving techniques, but these techniques typically require the cooperation of the access point to which the radio is connected. As a result, power consumption for the Wi-Fi radio can vary, depending on the capabilities of the connected access point.

To start measuring Wi-Fi radio power consumption, connect the system to a Wi-Fi access point that does not have public Internet connectivity. This helps to isolate the added power of the Wi-Fi connection from the added power of increased system activity due to an active Internet connection. You should use an instrumented system that is capable of measuring Wi-Fi module power in isolation to verify that Wi-Fi power consumption is much less than 15 milliwatts on average when connected to the Wi-Fi access point. Most Wi-Fi parts consume between 5 and 10 milliwatts when they are connected and power-save mode (PSM) is enabled.

Measurement scenario	Expected result	Power notes
Wi-Fi module power when connected to an access point without a public Internet connection.	Average power for the Wi-Fi module itself should be less than 15 milliwatts. When the Wi-Fi radio is connected, it does not have a constant power consumption due to beacon events. Measuring average power over an hour or more is critical.	Measurement should be performed while the system under test is on battery power. You should connect the Wi-Fi radio to several different brands of Wi-Fi access points to validate consistent power consumption. You should validate power consumption over several very long runs, including runs of up to 24 hours. Note that the Wi-Fi radio firmware and driver must be very mature to maintain connectivity for 24 hours. We recommend that you test connectivity for 24 hours with the system screen turned on (that is, outside of modern standby) before testing connectivity for 24 hours in modern standby.

After measuring Wi-Fi module power, you should validate the system for SoC activity during modern standby with Wi-Fi connected to the public Internet. For this scenario, the Wi-Fi access point should be connected to the public Internet and the system should remain in modern standby on battery power for several hours. The purpose of

the measurement is to verify that system activity stays within the expected levels (active no more than 10 percent of the time).

Measurement scenario	Expected result	Power notes
System activity is no more than 10 percent of the modern standby session when Wi-Fi is connected to the public Internet.	Place the system in modern standby for 4 hours with Wi-Fi connected to a public Internet connection. At the end of the test duration, wake the system and generate a SleepStudy report. The modern standby session should stay in the low-power state more than 90 percent of the time.	If the session has less than 90 percent low-power state time, use the Top Offenders table in a SleepStudy report to identify the active components. If the SleepStudy report is not helpful, capture a trace of the modern standby session, as explained in Capture and View a WPA Trace for Modern Standby Diagnostics. If you send a bug report to Microsoft for this problem, please include the SleepStudy-Report.html file and an ETW trace that is captured by using the instructions in Capture and View a WPA Trace for Modern Standby Diagnostics.

Overview of Modern Standby Testing and Diagnostics

Article • 03/22/2021

Enabling the low power consumption of Modern Standby systems and their ability to maintain network connectivity during sleep is a test and validation challenge for the system integrator. All components in the system—hardware and software—must work together to quickly turn power on and off while being able to maintain connection to the network.

This section is divided into two main categories: testing and diagnostics. The *Testing* section addresses test procedures to validate several Modern Standby scenarios. The *Diagnostics* section focuses on holistically verifying the performance and expected behavior of Modern Standby sessions in addition to drilling in on problem cases.

Verifying if a system is Modern Standby capable

Use the `powercfg /a` command to identify if a system supports Modern Standby. It will also identify whether the system is able to support network connectivity during sleep. The command produces the following output when the system supports Connected Standby:

Windows Command Prompt

```
The following sleep states are available on this system:
Standby (S0 Low Power Idle) Network Connected
```

If a system's NIC is not offload-capable, but the system reports in the FADT table that it supports S0 low power idle, it is using Disconnected Standby. In that case, the command produces the following output:

Windows Command Prompt

```
The following sleep states are available on this system:
Standby (S0 Low Power Idle) Network Disconnected
```

> ⓘ **Note**

The **Modern Standby SleepStudy report** also notes whether a standby session was connected or not.

For additional powercfg commands see Powercfg Command Line Options.

Power/performance targets

Article • 09/25/2020

The requirements for each system are dictated by the target market, workload, and form factor. OEMs should do their own modeling of their devices to determine the best power and performance targets.

Microsoft recommends the following performance goals:

- 1 second to resume from Standby.
- Software DRIPS achieved >= 80%
- Divergence between Software and Hardware DRIPS <= 10%

> **Note**
>
> In Windows 10 version 2004, Microsoft will not be enforcing the Modern Standby HLK requirement for 1 second recovery time ("instant on") if the device contains a Discrete GPU (dGPU).

> **Note**
>
> A device with rotational hard disk drive (HDD) as its primary boot drive is excused from exit latency requirement.

Tools available

Article • 03/21/2022

Several tools and test suites are available to evaluate a Modern Standby system. This page includes the summary information of each test suite. Additional platform specific tools might be available from silicon partners. Contact silicon partners for additional details.

`Powercfg /sleepstudy`

A powerful tool that can aid in the analysis of system behavior, and which components may be preventing a platform from reaching or maintaining its lowest power state. Modern standby sleep study contains more information on the SleepStudy Report.

> **Note**
>
> For diagnosing problems in Modern Standby,start with this SleepStudy report.

- **Availability:** Cmd tool available in Windows
- **When to use:**
  - To track overall system health, performance, and energy usage of a Modern Standby session.
  - To identify and diagnose issues during a Modern Standby session.

`Powercfg /a`

Reports the sleep states available on the computer. Reports reasons why sleep states are unavailable.

- **Availability:** Cmd tool available in Windows
- **When to use:** To verify the platform state enabled (CS vs. MS vs. S3)

Windows Performance Analyzer (WPA)

WPA displays graphs and data tables of Event Tracing for Windows (ETW) events that are recorded by Windows Performance Recorder (WPR), Xperf, or an assessment that is run in the Assessment Platform. WPA can open any event trace log (ETL) file for analysis.

- **Availability:** Windows Performance Analyzer (available in ADK and SDK kits)

- **When to use:** For more detailed analysis performance and DRIPS residency issues.

Hardware Lab Kit (HLK) tests

These tests are used to certify the quality of a Modern Standby device before shipment. More information on the HLK test suite can be found in the Windows Hardware Lab Kit. Some tests are required for certification, while others are not.

- **Availability:** HLK test suite
- **When to use:** To ensure Modern Standby devices have a good user experience spanning a variety of concerns including battery life, reliable entrance to modern sleep, latency resuming from modern sleep, and thermal behavior.

The required tests include:

- Check PM Profile
- Directed FX System Verification Test
- Lid Basic Requirement Test
- Modern Standby Basic Requirement Test on AC-power Source
- Modern Standby Basic Requirement Test on DC-power Source
- Modern Standby Check Thermal Zones
- MultiPhaseResume
- Processor Power Management

The optional tests include:

- Battery Validation - Drain Test
- Battery Validation - Drain Test Reset
- Directed FX Single Device Test
- Modern Standby Duration Test
- Modern Standby Stress with Driver Verifier's Concurrency Stress
- Modern Standby Wake on Voice Basic Test on AC-power Source
- Modern Standby Wake on Voice Basic Test on DC-power Source
- Runtime Power Focused Stress with Driver Verifier's Concurrency Stress

Windows Assessment and Deployment Kit (ADK)

These tests help system developers analyze the quality and performance of their systems. The test suite contains the following test to analyze the impact of software and

devices on the battery life of a system while it is in Modern Standby: Connected Standby energy efficiency.

- **Availability:** ADK test suite
- **When to use:** To measure the battery life of a system in Modern Standby and determine the time spent transitioning in and out of Modern Standby.

PwrTest

This command line tool can be used to stress test Modern Standby systems. For Modern Standby, designers can specifically make use of the PwrTest connected standby scenario.

- **Availability:** Available in Windows Desktop
- **When to use:** To run multiple iterations of tests, diagnose if a system is entering deep platform idle states, and determine if any software components are blocking the transition.

Powercfg command-line options

Article • 12/15/2021

Use powercfg.exe to control *power plans* - also called *power schemes* - to use the available sleep states, to control the power states of individual devices, and to analyze the system for common energy-efficiency and battery-life problems.

Syntax

Powercfg command lines use the following syntax:

powercfg */option* [*arguments*] [**/?**]

where *option* is one of the options listed in the following table, and *arguments* is one or more arguments that apply to the selected option. Including **/?** in a command line displays help for the specified option. Options and arguments are described in more detail later in this topic.

Command-line options

Option	Description
/?, -help	Displays information about command-line parameters.
/list, /L	Lists all power schemes.
/query, /Q	Displays the contents of a power scheme.
/change, /X	Modifies a setting value in the current power scheme.
/changename	Modifies the name and description of a power scheme.
/duplicatescheme	Duplicates a power scheme.
/delete, /D	Deletes a power scheme.
/deletesetting	Deletes a power setting.
/setactive, /S	Makes a power scheme active on the system.
/getactivescheme	Retrieves the currently active power scheme.
/setacvalueindex	Sets the value associated with a power setting while the system is powered by AC power.

Option	Description
/setdcvalueindex	Sets the value associated with a power setting while the system is powered by DC power.
/import	Imports all power settings from a file.
/export	Exports a power scheme to a file.
/aliases	Displays all aliases and their corresponding GUIDs.
/getsecuritydescriptor	Gets a security descriptor associated with a specified power setting, power scheme, or action.
/setsecuritydescriptor	Sets a security descriptor associated with a power setting, power scheme, or action.
/hibernate, /H	Enables and disables the hibernate feature.
/availablesleepstates, /A	Reports the sleep states available on the system.
/devicequery	Returns a list of devices that meet specified criteria.
/deviceenableawake	Enables a device to wake the system from a sleep state.
/devicedisablewake	Disables a device from waking the system from a sleep state.
/lastwake	Reports information about what woke the system from the last sleep transition.
/waketimers	Enumerates active wake timers.
/requests	Enumerates application and driver Power Requests.
/requestsoverride	Sets a Power Request override for a particular Process, Service, or Driver.
/energy	Analyzes the system for common energy-efficiency and battery life problems.
/batteryreport	Generates a report of battery usage.
/sleepstudy	Generates a diagnostic system power transition report.
/srumutil	Dumps Energy Estimation data from System Resource Usage Monitor (SRUM).
/systemsleepdiagnostics	Generates a diagnostic report of system sleep transitions.
/systempowerreport	Generates a diagnostic system power transition report.

Command-line option descriptions

The following sections describe Powercfg command-line options and arguments.

-help or /?

Displays information about command-line parameters.

Syntax:

```
powercfg /?
```

/list or /L

Lists all power schemes.

Syntax:

```
powercfg /list
```

/query or /Q

Displays the contents of the specified power scheme.

Syntax:

```
powercfg /query [scheme_GUID] [sub_GUID]
```

If neither the parameter *scheme_GUID* or *sub_GUID* are provided, the settings of the current active power scheme are displayed. If the parameter *sub_GUID* is not specified, all settings in the specified power scheme are displayed.

Arguments:

scheme_GUID

Specifies a power scheme GUID. Running **powercfg /list** returns a power scheme GUID.

sub_GUID

Specifies a power-setting subgroup GUID. A power setting subgroup GUID is returned by running **powercfg /query**.

Examples:

```
powercfg /query
powercfg /query 381b4222-f694-41f0-9685-ff5bb260df2e 238c9fa8-0aad-41ed-
83f4-97be242c8f20
```

/change or /X

Modifies a setting value in the current power scheme.

Syntax:

/change *setting value*

Arguments:

setting

Specifies one of the following options:

- monitor-timeout-ac
- monitor-timeout-dc
- disk-timeout-ac
- disk-timeout-dc
- standby-timeout-ac
- standby-timeout-dc
- hibernate-timeout-ac
- hibernate-timeout-dc

value

Specifies the new value, in minutes.

Examples:

```
powercfg /change monitor-timeout-ac 5
```

/changename

Modifies the name of a power scheme and optionally its description.

Syntax:

powercfg /changename *scheme_GUID * *name* [*description*]

Arguments:

scheme_GUID

Specifies a power scheme GUID. Running **powercfg /list** returns a power scheme GUID.

name

Specifies the power scheme's new name.

description

Specifies the power scheme's new description. If no description is specified, only the name is changed.

Examples:

```
powercfg /changename 381b4222-f694-41f0-9685-ff5bb260df2e "Customized
Balanced"
```

/duplicatescheme

Duplicates the specified power scheme. The resulting GUID which represents the new scheme is displayed.

Syntax:

powercfg /duplicatescheme *scheme_GUID* [*destination_GUID*]

Arguments:

scheme_GUID

Specifies a power scheme GUID. A power scheme GUID is returned by running **powercfg /list**.

destination_GUID

Specifies the new power scheme's GUID. If no GUID is specified, a new GUID is created.

Examples:

```
powercfg /duplicatescheme 381b4222-f694-41f0-9685-ff5bb260df2e
```

/delete or /D

Deletes the power scheme with the specified GUID.

Syntax:

powercfg /delete *scheme_GUID*

Arguments:

scheme_GUID

Specifies a power scheme GUID. A power scheme GUID is returned by running **powercfg /list**.

Examples:

```
powercfg /delete 381b4222-f694-41f0-9685-ff5bb260df2e
```

/deletesetting

Deletes a power setting.

Syntax:

powercfg /deletesetting *sub_GUID setting_GUID*

Arguments:

sub_GUID

Specifies a power setting subgroup GUID. A power setting subgroup GUID is returned by running **powercfg /query**.

setting_GUID

Specifies a power setting GUID. A power setting GUID is returned by running **powercfg /query**.

Examples:

```
powercfg /deletesetting 238c9fa8-0aad-41ed-83f4-97be242c8f20 29f6c1db-86da-48c5-9fdb-f2b67b1f44da
```

/setactive or /S

Makes the specified power scheme active on the system.

Syntax:

powercfg /setactive *scheme_GUID*

Arguments:

scheme_GUID

Specifies a power scheme GUID. A power scheme GUID is returned by running **powercfg /list**.

Examples:

```
powercfg /setactive 381b4222-f694-41f0-9685-ff5bb260df2e
```

/getactivescheme

Retrieves the currently active power scheme.

Syntax:

powercfg /getactivescheme

/setacvalueindex

Sets the value associated with a specified power setting while the system is powered by AC power.

Syntax:

powercfg /setacvalueindex *scheme_GUID sub_GUID setting_GUID setting_index*

Arguments:

scheme_GUID

Specifies a power scheme GUID. A power scheme GUID is returned by running **powercfg /list**.

sub_GUID

Specifies a power setting subgroup GUID. Running **powercfg /query** returns a power setting subgroup GUID.

setting_GUID

Specifies a power setting GUID. A power setting GUID is returned by running **powercfg /query**.

setting_index

Specifies which possible value this setting is set to. A list of possible values is returned by running **powercfg /query**.

Examples:

```
powercfg /setacvalueindex 381b4222-f694-41f0-9685-ff5bb260df2e 238c9fa8-
0aad-41ed-83f4-97be242c8f20 29f6c1db-86da-48c5-9fdb-f2b67b1f44da 0
```

/setdcvalueindex

Sets the value associated with a specified power setting while the system is powered by DC power.

Syntax:

powercfg /setdcvalueindex *scheme_GUID sub_GUID setting_GUID setting_index*

Arguments:

scheme_GUID

Specifies a power scheme GUID. A power scheme GUID is returned by running **powercfg /list**.

sub_GUID

Specifies a power setting subgroup GUID. A power setting subgroup GUID is returned by running **powercfg /query**.

setting_GUID

Specifies a power setting GUID. A power setting GUID is returned by running **powercfg /query**.

setting_index

Specifies which possible value this setting is set to. A list of possible values is returned by running **powercfg /query**.

Examples:

```
powercfg /setdcvalueindex 381b4222-f694-41f0-9685-ff5bb260df2e 238c9fa8-
0aad-41ed-83f4-97be242c8f20 29f6c1db-86da-48c5-9fdb-f2b67b1f44da 300
```

/import

Imports a power scheme from the specified file.

Syntax:

powercfg /import *file_name* [*GUID*]

Arguments:

file_name

Specifies a fully-qualified path to a file returned by running **powercfg /export**.

GUID

Specifies the GUID for the imported scheme. If no GUID is specified, a new GUID is created.

Examples:

```
powercfg /import c:\scheme.pow
```

/export

Exports a power scheme, represented by the specified GUID, to the specified file.

Syntax:

powercfg /export *file_name GUID*

Arguments:

file_name

Specifies a fully-qualified path to a destination file.

GUID

Specifies a power scheme GUID. A power scheme GUID is returned by running **powercfg /list**.

Examples:

```
powercfg /export c:\scheme.pow 381b4222-f694-41f0-9685-ff5bb260df2e
```

/aliases

Displays a list of aliases and their corresponding GUIDs. These aliases may be used instead of a GUID in any command.

Syntax:

powercfg /aliases

> ⓘ **Note**
>
> Some settings do not contain aliases. For a full list of GUIDs, use **powercfg /query**.

/getsecuritydescriptor

Gets the security descriptor associated with the specified power setting, power scheme, or action.

Syntax:

powercfg /getsecuritydescriptor *GUID* | *action*

Arguments:

GUID

Specifies a power scheme or a power setting GUID. A power scheme GUID is returned by running **powercfg /list**. A power setting GUID is returned by running **powercfg /query**.

action

Specifies one of the following actions:

- ActionSetActive
- ActionCreate
- ActionDefault

Examples:

```
powercfg /getsecuritydescriptor 381b4222-f694-41f0-9685-ff5bb260df2e<br />
powercfg /getsecuritydescriptor ActionSetActive
```

/setsecuritydescriptor

Sets a security descriptor associated with the specified power setting, power scheme, or action.

Syntax:

powercfg /setsecuritydescriptor *GUID* | *action SDDL*

Arguments:

GUID

Specifies a power scheme or a power setting GUID. A power scheme GUID is returned by running **powercfg /list**. A power setting GUID is returned by running **powercfg /query**.

action

Specifies one of the following actions:

- ActionSetActive
- ActionCreate
- ActionDefault

SDDL

Specifies a valid security descriptor string in SDDL format. An example SDDL string can be obtained by running **powercfg /getsecuritydescriptor**.

Examples:

```
powercfg /setsecuritydescriptor 381b4222-f694-41f0-9685-ff5bb260df2e
O:BAG:SYD:P(A;CI;KRKW;;;BU)(A;CI;KA;;;BA)(A;CI;KA;;;SY)(A;CI;KA;;;CO)<br />

powercfg /setsecuritydescriptor ActionSetActive O:BAG:SYD:P(A;CI;KR;;;BU)
(A;CI;KA;;;BA)(A;CI;KA;;;SY)(A;CI;KA;;;CO)
```

/hibernate or /H

Enables or disables the hibernate feature; also, sets the hiberfile size.

Syntax:

powercfg /hibernate

powercfg /hibernate [**on** | **off**]

powercfg /hibernate [/size *percent_size*]

powercfg /hibernate [/type *reduced* | *full*]

Arguments:

On

Enables the hibernate feature.

Off

Disables the hibernate feature.

/size *percent_size*

Specifies the desired hiberfile size as a percentage of the total memory size. The default size cannot be smaller than 50. This parameter also causes hibernation to be

enabled.

/type *reduced* | *full*

Specifies the desired hiberfile type. A reduced hiberfile only supports hiberboot.

> ⓘ **Note**
>
> A hiberfile that has a custom default size, or HiberFileSizePercent >= 40, is considered as a full hiberfile. HiberFileSizePercent is set in the registry in HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Power.
>
> To change the hiberfile type to reduced, the OS has to manage the default hiberfile size. To do this, run the following commands:
>
> **powercfg /hibernate /size 0**
>
> **powercfg /hibernate /type reduced**

Examples:

```
powercfg /hibernate off<br /> powercfg /hibernate /size 100<br /> powercfg /hibernate /type reduced
```

/availablesleepstates or /A

Reports the sleep states available on the system. Attempts to report reasons why sleep states are unavailable.

Syntax:

powercfg /availablesleepstates

/devicequery

Returns a list of devices that meet the specified criteria.

Syntax:

powercfg /devicequery *query_flag*

Arguments:

query_flag

Specifies one of the following criteria:

wake_from_S1_supported Returns all devices that support waking the system from a light sleep state.

wake_from_S2_supported Returns all devices that support waking the system from a deeper sleep state.

wake_from_S3_supported Returns all devices that support waking the system from the deepest sleep state.

wake_from_any Returns all devices that support waking the system from any sleep state.

S1_supported Lists devices supporting light sleep.

S2_supported Lists devices supporting deeper sleep.

S3_supported Lists devices supporting deepest sleep.

S4_supported List devices supporting hibernation.

wake_programmable Lists devices that are user-configurable to wake the system from a sleep state.

wake_armed Lists devices that are currently configured to wake the system from any sleep state.

all_devices Returns all devices present in the system.

Examples:

```
powercfg /devicequery wake_armed
```

/deviceenableawake

Enables the specified device to wake the system from a sleep state.

Syntax:

powercfg /deviceenableawake *device_name*

Arguments:

device_name

Specifies a device. This device name may be retrieved using **powercfg /devicequery wake_programmable**.

Examples:

```
powercfg /deviceenableawake &quot;Microsoft USB IntelliMouse Optical&quot;
```

/devicedisablewake

Disables the specified device from waking the system from a sleep state.

Syntax:

powercfg /devicedisablewake *device_name*

Arguments:

device_name

Specifies a device. This device name may be retrieved using **powercfg /devicequery wake_armed**.

Examples:

```
powercfg /devicedisablewake &quot;Microsoft USB IntelliMouse Optical&quot;
```

/lastwake

Reports information about what woke the system from the last sleep transition.

Syntax:

powercfg /lastwake

/waketimers

Enumerates the active wake timers. If enabled, the expiration of a wake timer wakes the system from sleep and hibernate states.

Syntax:

```
powercfg /waketimers
```

/requests

Enumerates application and driver Power Requests. Power Requests prevent the computer from automatically powering off the display or entering a low-power sleep mode.

Syntax:

```
powercfg /requests
```

/requestsoverride

Sets a Power Request override for a particular process, service, or driver. If no parameters are specified, this command displays the current list of Power Request overrides.

Syntax:

```
powercfg /requestsoverride [caller_type name request]
```

Arguments:

Caller_type

Specifies one of the following caller types: **process**, **service**, **driver**. This is obtained by running **powercfg /requests**.

name

Specifies the caller name. This is the name returned by running **powercfg /requests**.

request

Specifies one or more of the following Power Request types:

- Display
- System
- Awaymode

Examples:

```
powercfg /requestsoverride process wmplayer.exe display system
```

/energy

Analyzes the system for common energy-efficiency and battery-life problems and generates a report, an HTML file, in the current path.

Syntax:

powercfg /energy [/output *file_name*] [/xml] [/duration *seconds*]

powercfg /energy /trace [/d *file_path*] [/xml] [/duration *seconds*]

The **/energy** option should be used when the computer is idle and has no open programs or documents.

Arguments:

/output *file_name*

Specify the path and file name to store the energy report HTML or XML file.

/xml

Formats the report file as XML.

/duration *seconds*

Specifies the number of seconds to observe system behavior. Default is 60 seconds.

/trace

Records system behavior and does not perform analysis. Trace files are generated in the current path unless the **/D** parameter is specified.

/d *file_path*

Specify the directory to store trace data. May only be used with the **/trace** parameter.

Examples:

```
```

```
powercfg /energy<br />
powercfg /energy /output &amp;quot;longtrace.html&amp;quot; /duration 120
```

/batteryreport

Generates a report of battery usage characteristics over the lifetime of the system. Running **powercfg /batteryreport** generates an HTML report file in the current path.

Syntax:

powercfg /batteryreport [/output *file_name*] [/xml]

powercfg /batteryreport [/duration *days*]

Arguments:

/output *file_name*

Specify the path and file name to store the battery report HTML.

/output *file_name* /xml

Formats the battery report file as XML.

/duration *days*

Specifies the number of days to analyze for the report.

Examples:

```
powercfg /batteryreport /output &amp;quot;batteryreport.html&amp;quot;<br />
powercfg /batteryreport /duration 4
```

/sleepstudy

Generates a diagnostic report of modern standby quality over the last three days on the system. The report is a file that is saved in the current path.

Syntax:

powercfg /sleepstudy [/output *file_name*] [/xml]

powercfg /sleepstudy [/duration *days*]

powercfg /sleepstudy [/transformxmL *file_name.xml*] [/output *file_name.html*]

Arguments:

/output *file_name*

Specify the path and file name to store the Sleepstudy report HTML.

/output *file_name* /xml

Formats the Sleepstudy report file as XML.

/duration *days*

Specifies the number of days to analyze for the report.

/transformxml *file_name.xml* /output *file_name.html*

Transforms the Sleepstudy report from XML to HTML.

Examples:

```
powercfg /sleepstudy /output &quot;sleepstudy.html&quot;<br />
powercfg /sleepstudy /duration 7
```

/srumutil

Enumerates the entire Energy Estimation data from the System Resource Usage Monitor (SRUM) in an XML or CSV file.

Syntax:

powercfg /srumutil [/output *file_name*] [/xml] [/csv]

Arguments:

/output *file_name*

Specify the path and file name to store the SRUM data.

/output *file_name* /xml

Formats the file as XML.

/output *file_name* /csv

Formats the file as CSV.

Examples:

```
powercfg /batteryreport /output &amp;quot;srumreport.xml&amp;quot; /xml
```

/systemsleepdiagnostics

Generates a report of intervals when the user was not present over the last three days on the system, and if the system went to sleep. This option generates a report, an HTML file, in the current path.

This command requires administrator privileges and must be executed from an elevated command prompt.

Syntax:

powercfg /systemsleepdiagnostics [/output *file_name*] [/xml]

Arguments:

/output *file_name*

Specifies the path and file name of the diagnostics report.

/xml

Save the report as an XML file.

/duration *days*

Specifies the number of days to analyze for the report.

/transformxml *file_name*

Produces a report in HTML from a report that was previously created in XML.

Examples:

```
powercfg /systemsleepdiagnostics<br/>
powercfg /systemsleepdiagnostics /output &amp;quot;system-sleep-
diagnostics.html&amp;quot;<br/>
powercfg /systemsleepdiagnostics /output &amp;quot;system-sleep-
```

```
diagnostics.xml&amp;quot; /XML<br/>
powercfg /systemsleepdiagnostics /transformxml &amp;quot;system-sleep-
diagnostics.xml&amp;quot;
```

/systempowerreport or /spr

Generates a report of system power transitions over the last three days on the system, including connected standby power efficiency. This option generates a report, an HTML file, in the current path.

This command requires administrator privileges and must be executed from an elevated command prompt.

Syntax:

powercfg /getsecuritydescriptor *GUID* | *action*

Arguments:

/output *file_name*

Specifies the path and file name of the diagnostics report.

/xml

Save the report as an XML file.

/duration *days*

Specifies the number of days to analyze for the report.

/transformxml *file_name*

Produces a report in HTML from a report that was previously created in XML.

Examples:

```
powercfg /systempowerreport<br/>
powercfg /systempowerreport /output &amp;quot;sleepstudy.html&amp;quot;<br/>
powercfg /systempowerreport /output &amp;quot;sleepstudy.xml&amp;quot; /XML<br/>
powercfg /systempowerreport /transformxml &amp;quot;sleepstudy.xml&amp;quot;
```

Overlay Scheme and PPM Profile Support

Overlay power schemes and PPM profiles can now be customized through powercfg.exe. It is important to note that overlay schemes are now limited to customizing settings that affect performance versus power savings tradeoff. This is currently related to settings under the PPM and Graphics power settings subgroups (with aliases SUB_PROCESSOR and SUB_GRAPHICS in powercfg). Attempts to write to other subgroups under overlay schemes will result in an error message.

Reading from overlay schemes

The powercfg commands used earlier to read power schemes now support overlay schemes as well for reads and writes.

Syntax:

powercfg /q *overlay_scheme_alias subgroup_alias setting_alias*

All arguments after the /q flag are optional. If the setting alias is not specified, all settings under the specified overlay scheme and subgroup will be enumerated. If the subgroup is not specified, then all settings for all subgroups under the specified overlay scheme will be enumerated. If the overlay scheme is not specified, then it will be assumed to be the currently active overlay scheme (if active) or the current power scheme (if no overlay is active).

Writing to overlay schemes

The commands setacvalueindex and setdcvalueindex now support overlay schemes as well.

Syntax:

powercfg /setacvalueindex *overlay_scheme_alias subgroup_alias setting_alias value*

powercfg /setdcvalueindex *overlay_scheme_alias subgroup_alias setting_alias value*

Reading from PPM profile

The commands are similar to that of overlay schemes and power schemes, except that they use the /qp flag.

Syntax:

powercfg /queryprofile *overlay_or_power_scheme_alias profile_alias setting_alias*

powercfg /qp *overlay_or_power_scheme_alias profile_alias setting_alias*

PPM profile aliases are visible by running the same powercfg /aliasesh command. Support for missing arguments is provided, and the behavior is similar to when arguments are missing and the /q flag is used.

Writing to PPM profile

For writing to PPM profiles, the /setacprofileindex and /setdcprofileindex commands can be used.

Syntax:

powercfg /setacprofileindex *overlay_or_scheme_alias profile_alias setting_alias value*

powercfg /setdcprofileindex *overlay_or_scheme_alias profile_alias setting_alias value*

Enumerating non-empty PPM Profiles

For enumerating PPM profiles which have at least one power setting value explicitly set.

Syntax:

powercfg /listprofiles

powercfg /lp

Provisioning XML Generation Support

Powercfg now supports automatically generating a provisioning XML file that can be used as an input to Windows Configuration Designer in order to generate a provisioning package (.ppkg) that contains the customized settings from a device under test. This file contains all settings on the device with the "RUNTIME_OVERRIDE" altitude value.

Syntax:

powercfg /pxml /output *output_file_path*

powercfg /pxml /output *output_file_path* /version *version_number* /name *package_name* /id *GUID* /owner *OwnerType_value*

Required Arguments:

/output_file_path: Specifies the location and name of the generated XML.

Optional Arguments:

/version: Optionally specifies the value of the "Version" field in the generated XML. Default: *1.0*

/name: Optionally specifies the value of the "Name" field in the generated XML. Default: *CustomOEM.Power.Settings.Control*

/id: Optionally specifies a GUID string that is used in the "ID" field in the generated XML. Default: *new GUID is generated*

/owner: Optionally specified the value of the "OwnerType" field in the generated XML. Default: *OEM*

Where to start with Modern Standby testing

Article • 03/17/2021

Testing and validating the operation of a Modern Standby PC is a critical effort for the system integrator. Modern Standby involves all hardware and software components on the system, and requires special attention to the core silicon or System on a Chip (SoC), networking devices (for example, Wi-Fi), and peripheral devices connected to the pins of the SoC chip.

Before testing, make sure that the required hardware connections are in place. The target platform must have a functioning kernel debugger connection. The following is the recommended testing scenario order:

1. **Validate entry and exit from Modern Standby**

   The first test scenarios to validate are entry and exit from Modern Standby. These transitions are easily initiated by using the system power button to turn the display on and off.

2. **Validate airplane mode**

   Run the platform in airplane mode for increasingly long durations, and focus on reducing platform power consumption during the idle periods that occur in Modern Standby. For more information about the test cases and methodology for these scenarios, see Modern Standby basic test scenarios.

3. **Validate connected Wi-Fi and mobile broadband scenarios**

   Focus on scenarios in which Wi-Fi is connected and background tasks are running. Make sure that Wi-Fi power can be measured in isolation from system power to verify that Wi-Fi operates in the expected low-power mode. Validation concludes with mobile broadband (cellular) testing. For more information about the test cases and methodology for these scenarios, see Modern Standby Wi-Fi-connected scenarios.

After any driver or firmware fixes, it is important to conduct regression testing.

Finally, system designers are recommended to also incorporate a self-host program into their validation program for a Modern Standby PC. A self-host program enables local engineering and engineering-management teams to use reference designs or early retail designs for daily work and home use. The goal of the self-host program for

Modern Standby is to find functional and power consumption issues that are otherwise not identified by planned test execution.

Please review tools available for Modern Standby diagnostics and validation.

Modern Standby testing prerequisites

Article • 09/25/2020

The System on a Chip (SoC) platform should meet a set of basic software and hardware requirements before testing and validating Modern Standby operation. These requirements help to ensure that any problems that might arise are quickly identified and easily debugged.

Physical setup and equipment

Effective Modern Standby testing cannot be accomplished without first ensuring that the system under test has the basic physical setup that is required to capture diagnostic information. The setup for this system should include the equipment listed in the following table.

Equipment	Purpose	Physical setup
Kernel debug board and cable	Supports Windows kernel debugging. Consult with the SoC vendor to determine what type of debugging is supported and what equipment is necessary. Some systems require a specialized debugging board.	Connect the kernel debugger between a host PC and the system under test.
Host PC(s)	Serves as the host for kernel and hardware debugging. The same or separate PCs can be used for kernel and hardware debugging.	Connect the PC(s) to the kernel and hardware debuggers.
Power meter	A power meter is required to measure power consumption on a power-instrumented system. Consult the SoC vendor to determine the appropriate device to measure power.	When measuring the power floor, connect the power meter to the platform that is being tested.

Equipment	Purpose	Physical setup
SIM card	A SIM card is required to test mobile broadband (MBB) connectivity during Modern Standby.	When testing MBB operation, plug the SIM card into the SIM card slot.

System hardware requirements

The system under test must be built of hardware components that are suitable for a Modern Standby PC and that can operate at low power during standby. Work closely with the SoC and power-management IC (PMIC) vendors to identify the appropriate version of SoC and PMIC to use for Modern Standby testing. All off-SoC device components—such as Wi-Fi, Bluetooth, and near-field communication (NFC) devices—must also support low power.

Before testing Modern Standby, verify the following:

- The system is built with the appropriate version of SoC for low-power/Modern Standby testing.
- The system is built with the appropriate version of PMIC for low-power/Modern Standby testing.
- All off-SoC device components support low-power operation and are wired to switchable power planes.

Driver and firmware requirements

Drivers and firmware play a critical role in Modern Standby operations and are therefore an important part of Modern Standby testing.

The system must have the appropriate drivers and firmware installed to allow the system to enter low-power states. The easiest way to verify that all firmware and drivers are properly installed is to open Device Manager and check that no devices have active problem codes (yellow ! symbols) and that no devices are missing drivers.

Do not proceed with Modern Standby testing unless all of the following is true:

- All system and device firmware is properly installed.
- All device drivers are properly installed.
- All devices in Device Manager do not show a problem code (yellow !).
- There are no missing drivers for devices in Device Manager.

Software requirements

The system must have a stable and functioning graphics driver before testing Modern Standby. Similarly, the Wi-Fi and mobile broadband (MBB) devices must be capable of functioning correctly when the screen is on before they can be properly tested for low-power and Modern Standby operation.

Before starting Modern Standby testing, make sure the system has the following:

- A stable graphics driver.
- Reliable Wi-Fi connectivity when the screen is on.
- Reliable MBB connectivity when the screen is on (if the system is equipped with MBB).

Modern Standby Basic Test Scenarios

Article • 07/08/2024

Stabilize basic functionality of Modern Standby before proceeding to more advanced test scenarios, such as those involving an active Wi-Fi connection. First, validate entry to and exit from Modern Standby. Then, validate airplane mode and audio playback.

Required System Configuration

Ensure the following configurations are correct before performing any of the tests below:

- Factory image is installed on the system.
- All drivers are loaded in Device Manager.
- Factory Microsoft Store apps are installed.
- System is running on battery power.
- All Windows updates are installed.

Basic Entry and Exit Testing

Begin all testing by ensuring that the system can reliably enter Modern Standby and then exit Modern Standby after a variable duration. To initiate entry to Modern Standby, press the power button, close the lid, or select Sleep from the power button in the Settings flyout. These actions should instantly power off the display. To wake up the system, simply press the power button or open the lid of the system. These actions should instantly power on the display.

If the system is connected to a USB mouse or keyboard, move the mouse or press any key to generate a wake event and turn on the display.

The following table enumerates the Modern Standby entry and exit scenarios. All Modern Standby systems should be tested for expected behavior in each scenario.

- **Test area:** Entry to and exit from Modern Standby.
- **Purpose:** Ensure the system can reliably and quickly enter and exit Modern Standby.
- **Additional system configuration requirements:**
  - None

Expand table

Test scenario	Expected result	Troubleshooting notes
The system can enter Modern Standby using the following methods: • Press the power button. • Select Sleep from the operating system settings. • Close the lid (if applicable).	The system instantly transitions into Modern Standby (display powers off). No audio noises are generated.	If the system does not enter Modern Standby, a trace should be taken to diagnose the problem. See Capture and View a WPA Trace for Modern Standby Diagnostics for instructions on how to capture a trace and analyze the information.
The system can enter Modern Standby using the following method: • Idle out	The system transitions into display powers off. The time this takes is the sleep timeout minus the display timeout settings. (Audio player may still playback with Screen Off until pressing power button).	If the system does not enter Modern Standby when user specifically press power button to sleep, a trace should be taken to diagnose the problem. See Capture and View a WPA Trace for Modern Standby Diagnostics for instructions on how to capture a trace and analyze the information.
The system can exit Modern Standby using the following methods: • Press the power button. • Open the lid (if applicable). • Use a USB mouse or keyboard. • Plugging in AC	The system instantly transitions out of Modern Standby (display powers on). Most failure cases in this category are the result of the system being unable to resume from Modern Standby. To debug, enable both the **kernel debugger** and the **hardware debugger** on the system to catch software crashes and hardware hard hangs, which can prevent the system from resuming.	

The Windows Hardware Lab Kit (HLK) includes Modern Standby tests that exercise entering and exiting Modern Standby on a system. Running the following tests will further validate the readiness of the system:

- Modern Standby basic verification test on AC-power
- Modern Standby basic verification test on DC-power

Basic Airplane Mode Testing

After verifying that the system can reliably enter and exit Modern Standby, validate system behavior during Modern Standby.

Modern Standby battery life is directly impacted by hardware and software activity. The goal of basic airplane mode testing is to identify activities that should not occur during Modern Standby and resolve them. These activities might be a result of software in the OEM pre-installation or unexpected hardware interrupts.

Start airplane mode testing with a one-hour Modern Standby session with airplane mode enabled. After resuming the system from Modern Standby, use SleepStudy to review the activities during the session and the amount of time that the system was in a low-power state.

- **Test area:** Airplane mode low-power state time.
- **Purpose:** Ensure that the basic activity level of the system is minimal (less than three percent of active time) with the factory image and apps installed.
- **Additional system configuration requirements:**
  - System is put into airplane mode.

Expand table

Test scenario	Expected result	Troubleshooting notes
Stay in Modern Standby in airplane mode for at least one hour.	Using SleepStudy, observe that the system is in the lowest power state equal to or greater than **80 percent** of the time.	SleepStudy helps identify which software or device component(s) are active, preventing the system from entering the lowest power state. Contact the owner of the offending component for debugging tips and next steps.

To eliminate noise from system resource loading on boot, wait at least 5 minutes after system boot before testing. As a general guideline, all Modern Standby testing should be performed on a system running on DC power so that the power impact can be measured.

Basic Audio Playback Testing

⊗ **Caution**

Starting from Windows 11 version 24H2, audio playback is not supported when standby is entered explicitly (power button press, lid close, or through the start menu). Audio playback is only supported when idling to screen off.

Audio playback during Modern Standby enables the system to play music while screen off. The system is expected to consume less power when playing music in Modern Standby than when playing music with the display powered on. The goal of audio playback testing is to verify that the system can play sound through the internal speakers and headphones, and that the user can adjust the volume using the volume buttons during Modern Standby in a low-power state.

Start with a system that is un-muted and use the inbox Music app to play a 256-kilobits/second .mp3 track. Put the system into Modern Standby by idling to standby. Make sure that the sound continues to play through the speakers and that the volume can be adjusted through the volume buttons while the screen is powered off. Also, try plugging in headphones to ensure that the sound is transferred from the internal speakers to the headphones.

Not all USB, Bluetooth, HDMI, and other external devices support low-power audio. Low-power audio playback depends on the hardware and software support of the audio device.

For audio devices that do not support low-power playback, audio playback will continue after the power button is pressed, but at an increased power consumption compared to output through audio devices that support low-power playback.

Support for low-power audio for USB audio was added in Windows Version 1903. Bluetooth A2DP low-power audio is available starting from Windows 11 version 21H2. In Windows 11 version 24H2 and later, audio is supported when idling to screen off and will not play when standby is entered explicitly (power button press, lid close, or through the start menu).

All audio devices are expected to be able to play during Modern Standby using supported apps. The power consumption will depend on the hardware and driver support.

Expand table

Test scenario	Expected result	Troubleshooting notes
When idling to Modern Standby, the system can play	The system continues to play sound through the speakers/headphones, and the volume can be adjusted using the volume buttons while the	If the system fails to enter a low-power state, a trace is needed to diagnose the problem. For information about how to capture

Test scenario	Expected result	Troubleshooting notes
audio, and sound plays from: • Internal speakers • Headphones • Bluetooth (If the hardware and drivers support low-power audio playback) • USB (If the hardware and drivers support low-power audio playback)	screen is powered off. Ensure that the system is consuming the expected power for low-power audio. Consult with the System on a Chip (SoC) vendor for the expected power consumption for low-power audio.	a trace and analyze the information, see Capture and View a WPA Trace for Modern Standby Diagnostics. If the audio stops when the screen is powered off, this usually means the music app does not have the background transfer API implemented correctly.

Testing for low-power audio playback must be done on a system running on battery power so that the power impact can be accurately measured.

Modern standby Wi-Fi-connected scenarios

Article • 11/05/2020

The goal of Wi-Fi-connected testing is to identify activities that should not be happening during modern standby and resolve them. The activities might be a result of network activities, real-time notifications, or unexpected hardware interrupts.

After the system can reliably enter and exit modern standby and can achieve good results in modern standby with airplane mode on, you can validate modern standby system behaviors with Wi-Fi connected.

To prepare for WiFi-connected testing, the following two prerequisites must be met:

- The system has a reliable Wi-Fi connection. The best way to assess this is to test the Wi-Fi connection during active use to make sure that the connection is stable and signal/speed is reasonably good.
- The Wi-Fi device is compatible with the NDIS 6.3 feature to support wake on LAN (WoL) patterns, protocol offloads, and D0 packet coalescing. These capabilities are required so that the SoC can enter low-power states while the Wi-Fi device maintains connectivity.

The list of Wi-Fi-connected scenarios includes connectivity to communications apps and downloads from the Internet. These scenarios should be tested serially to focus on issues that might fix similar issues in other scenarios. The Windows Hardware Certification Kit (HCK) includes tests that help validate basic NDIS 6.3 functionality for modern standby. There are individual tests for each type of network adapter, including Wi-Fi, CDMA MBB, GSM MBB, and wired LAN. Please see the test details information for each test at Device.Network Testing.

We recommend that each network adapter in the system pass the following tests before you proceed to system-level connectivity testing for modern standby:

- WLAN Connected Standby End to End - Basic
- Win.MBN.CDMA.TestCSConnectivity
- Win.MBN.GSM.TestCSConnectivity
- LAN CS Test - IPv4 Basic

Modern standby push mail

Testing the push mail scenario ensures that the system can receive emails via the Mail app while the system is in modern standby. This allows users to receive the most up-to-date information immediately after the system resumes.

We recommend that you create a Microsoft account for testing purposes. You can use this account to set up the system to receive emails through the Mail app. Be sure that the system is running on AC power, then put it into modern standby. While the system is in modern standby, send an email to the account a few times. Wake up the system from modern standby and verify that all the emails are received. If Mail is configured to be on the lock screen, you should also see the badge updates occur just as the screen is powered on.

- **Test area:** Push mail during modern standby.
- **Purpose:** Ensure that push mail is functional during modern standby.
- **System configuration:**
  - Factory image is installed on the system.
  - All drivers are loaded in Device Manager.
  - Factory Microsoft Store apps are installed.
  - System is running on AC power.
  - System is connected to a Wi-Fi access point (AP) that has Internet connectivity.
  - Mail app is configured with a test Microsoft account.
  - In Background Apps, Mail is set to **On**.
  - In Battery Usage:
    - Mail is set to **Allow the app to run background tasks**.
    - Mail is not set to **Let Windows decide...** or **Reduce the work...**.
  - Email operation is first validated with the display on.

Test scenario	Expected result	Troubleshooting notes
The system can receive mail notifications while in modern standby.	If the Mail app is configured to be on the lock screen, updates should be seen on the lock screen immediately on resume from modern standby. Ensure all emails are received.	When a notification is not received, a trace is needed to diagnose the problem. For information about how to capture a trace and analyze the information, see Capture and View a WPA Trace for Modern Standby Diagnostics.

The Windows Hardware Certification Kit (HLK) includes modern standby tests that exercise entering and exiting modern standby on a system. Running the following tests will further validate the readiness of the system:

- Modern standby basic verification test on AC-power
- Modern standby basic verification test on DC-power

Modern standby instant messaging and video chat

NOTE: Skype no longer supports this functionality starting with version 8.61.

Instant-message and video-chat notifications are important Wi-Fi-connected scenarios. The goal of testing this scenario is to ensure users can receive instant messages and video calls in real-time while the system is in modern standby.

You can test the instant messaging scenario with Skype by putting the system into modern standby and sending an instant message to the test Microsoft account. You should observe that the system powers on the display when an instant message is received and shows an instant message toast notification on the lock screen.

After you verify that instant message notifications are working, try to make a video call and an audio conference call. When the system is in modern standby, make a video call to the Skype test account. You should observe that the system powers on the display and shows a ringing toast notification at the upper right hand corner. If you do not pick up the call and let it finish ringing, the system will immediately go back into modern standby.

When the system wakes from modern standby, make sure that all notifications are received as expected. When an instant message or a call is missed, the icon on the lock screen will show the number of missed messages and calls.

The instant messaging and video chat scenarios are enumerated in the following table. All modern standby systems should be tested for expected behavior for each scenario that is listed in the table.

- **Test area:** Instant messaging (IM) and video chat during modern standby.
- **Purpose:** Ensure that real-time incoming functionality is operational during modern standby.
- **System configuration:**
  - Factory image installed on the system.
  - All drivers loaded in Device Manager.
  - Factory Microsoft Store apps are installed.
  - System is running on battery power.
  - System is connected to a Wi-Fi AP with Internet connectivity.
  - Skype app configured with test Microsoft account.
  - Skype IM and video chat are validated to work with the screen turned on (outside of modern standby).

- Skype has been configured to run in the background on the *Battery Settings* page

Test scenario	Expected result	Troubleshooting notes
System can receive IM notifications during modern standby.	When the system wakes from modern standby, check that all IM messages have been received and that the badge on the lock screen is updated appropriately.	If a notification is not received, a trace is needed to diagnose the problem. See Capture and View a WPA Trace for Modern Standby Diagnostics for information on how to capture a trace and analyze the information.
System can receive video call and audio conference call notifications during modern standby.	Ensure that all calls received during modern standby are accompanied by notifications that do the following: • Turn on the display to show a toast in the upper right-hand corner. If the call is missed, the icon on the lock screen will show the missed call when the system wakes from modern standby.	If a notification is not received, a trace is needed to diagnose the problem. See Capture and View a WPA Trace for Modern Standby Diagnostics for information on how to capture a trace and analyze the information.

Modern standby background download

Background download through a Wi-Fi connection is an important modern standby scenario to test. This scenario includes downloads from Windows Update for critical updates, and downloads of music and movies while the system stays in modern standby.

The goal of testing background download is to make sure that download is done in the background and does not continuously block the system from entering the lowest power state during modern standby. Note that background download is allowed to continuously block the system from using the lowest power state when the system is running on AC power. Testing background download on battery is required.

To test background downloads, it is best to test using a large file. You should also ensure the system is running on battery (not connected to AC) to get the correct behavior since systems on AC have special download policies to allow downloads to continue uninterrupted.

One way to test background download is use the inbox Video app to kick off a large download (over 3 gigabytes), and then put the system into modern standby. Windows permits the download to run through an activator in the Resiliency phase. Please see Prepare software for Modern Sleep for a conceptual overview of this software flow.

The key observation in the test is to verify that the large download does not prevent the system from entering Modern Sleep.

On resume from modern standby, you can run a SleepStudy report to view the time when the system was in Modern Sleep and, more specifically, DRIPS, to determine whether the background download was blocking the system from entering low power. For the best testing scenaio, the modern standby session should extend for at least one hour.

- Test area: Background file download during modern standby.
- Purpose: Ensure that a background download is performed at low-power operation during a modern standby session.
- System configuration:
  - Factory image installed on the system.
  - All drivers loaded in Device Manager.
  - Factory Microsoft Store apps are installed.
  - System is running on battery power.
  - System is configured with a test Microsoft account for inbox Video app access.

Test scenario	Expected result	Troubleshooting notes
Background downloads	A large file download should not continuously prevent the system from entering the lowest power state after the first 20 minutes.	With the exception of Windows Update, which can be active for a long period of time due to downloads of critical/security updates, all other downloads should not be active beyond the initial 20 minutes of the modern standby session. You can use SleepStudy to check whether the large file download prevented the system from entering the lowest power state for the entirety of the session.

Modern standby roaming connectivity

The most basic roaming connectivity scenario involves functional validation of network list offload (NLO). NLO is the capability of the Wi-Fi device to automatically roam between previously associated Wi-Fi networks, without the involvement of Windows running on the SoC.

NLO starts with Windows tracking the list of previously associated Wi-Fi networks. When the system transitions into modern standby, Windows communicates the list of previously associated Wi-Fi networks to the Wi-Fi device.

If connectivity to the associated Wi-Fi network is lost, the Wi-Fi device periodically searches for one of the previously associated Wi-Fi networks while the device operates in a very low power mode. The periodic scans for a previous network are performed without waking the SoC so that the SoC can stay in a low-power mode. When the Wi-Fi device detects a previously used network, it connects to the network and wakes the SoC. Windows completes the connection, including obtaining an IP address and re-establishing system connections, including Windows Notification Services (WNS).

Start testing Wi-Fi roaming by setting up two Wi-Fi networks (A and B) with public Internet connections. Each Wi-Fi network should have a different SSID and be sufficiently distant so that the system cannot see both networks at the same time. Manually move the system near each Wi-Fi access point and connect to each network for a few minutes with the screen on and validate that Internet connectivity is functional. After the system has been demonstrated to connect to each Wi-Fi network reliably, you can start testing Wi-Fi roaming in modern standby.

Modern standby Wi-Fi roaming

The most basic roaming connectivity scenario involves functional validation of network list offload (NLO). NLO is the capability of the Wi-Fi device to automatically roam between previously associated Wi-Fi networks, without the involvement of Windows running on the SoC.

NLO starts with Windows tracking the list of previously associated Wi-Fi networks. When the system transitions into modern standby, Windows communicates the list of previously associated Wi-Fi networks to the Wi-Fi device.

If connectivity to the associated Wi-Fi network is lost, the Wi-Fi device periodically searches for one of the previously associated Wi-Fi networks while the device operates in a very low power mode. The periodic scans for a previous network are performed without waking the SoC so that the SoC can stay in a low-power mode. When the Wi-Fi device detects a previously used network, it connects to the network and wakes the SoC.

Windows completes the connection, including obtaining an IP address and re-establishing system connections, including Windows Notification Services (WNS).

Start testing Wi-Fi roaming by setting up two Wi-Fi networks (A and B) with public Internet connections. Each Wi-Fi network should have a different SSID and be sufficiently distant so that the system cannot see both networks at the same time. Manually move the system near each Wi-Fi access point and connect to each network for a few minutes with the screen on and validate that Internet connectivity is functional. After the system has been demonstrated to connect to each Wi-Fi network reliably, you can start testing Wi-Fi roaming in modern standby.

- **Test area:**Wi-Fi roaming operation during modern standby.
- **Purpose:**Ensure that the system will automatically connect to previously used Wi-Fi networks during modern standby as the user moves the system between home, work, and coffee shops.
- **System configuration:**
  - Factory image is installed on the system.
  - All drivers are loaded in Device Manager.
  - Factory Microsoft Store apps are installed.
  - System is running on battery power.
  - Wi-Fi connectivity is validated to work by connecting to the Internet while the screen is turned on.

Test scenario	Expected result	Troubleshooting notes

Test scenario	Expected result	Troubleshooting notes
Wi-Fi connectivity is removed and the system automatically falls over to MBB while in modern standby.	Connect to the Wi-Fi network with MBB powered on, place the system into modern standby, and test push mail and Skype. Mail and Skype calls should be received. Power down the Wi-Fi access point and wait a few minutes for the system to automatically connect to the cellular network. Test push mail and Skype. Mail and Skype calls should be received. Power up the Wi-Fi access point again and wait a few minutes for the system to re-connect to Wi-Fi. Test push mail and Skype. Mail and Skype calls should be received.	Make sure you first test Wi-Fi roaming in insolation before testing MBB failover. Ensure that the system has a working cellular data connection. If any part of the test fails, first have the Wi-Fi and MBB vendors help you verify that the device firmware has both WoL pattern and NLO capability enabled.

The test engineer for Wi-Fi roaming should build a close working relationship with the Wi-Fi vendor's support staff. Errors in this test should first be discussed with the Wi-Fi firmware developer.

A common problem is that the Wi-Fi device generates spurious SoC wake events because of false-positive NLO network detection events. This condition appears in SleepStudy as extra Wi-Fi device active time—typically, much more active time than any other PDC activator (such as BI, Windows Update, or WNS).

Modern standby Wi-Fi to MBB failover

During modern standby, if Wi-Fi connectivity is lost, Windows automatically transitions to an available MBB connection. This behavior enables the system to remain constantly connected during modern standby. However, Windows will favor the cheaper, faster, and (typically) more power-efficient Wi-Fi connection over a cellular-based connection.

MBB and cellular connectivity are highly influenced by signal strength and proximity to a cellular tower. Make sure that the system under test has good cellular network connectivity before you test this scenario.

You should fully validate basic Wi-Fi roaming before you try to validate failover from Wi-Fi to MBB. The same Wi-Fi NLO technology that enables basic Wi-Fi roaming is critical to the operation of Wi-Fi to MBB failover.

The system under test should be configured to have an active cellular data plan and SIM card. The test setup should include a Wi-Fi access point with public Internet access. We recommend that you configure the system under test with a Microsoft account, and set up an additional device that is configured to initiate email and Skype calls.

Start the Wi-Fi to MBB failover testing by confirming that the system is connected to the Wi-Fi access point and has MBB access. You should validate that both network types are functional with the screen on. To do this, connect to Wi-Fi with the MBB radio off, and then connect to MBB with the Wi-Fi radio off. For each network connection, you should connect to the Internet and browse the web to verify IP connectivity.

Next, put the system in modern standby with both Wi-Fi and MBB radios turned on and the system connected to the Wi-Fi network. Use a Wi-Fi connection to test push mail and Skype during modern standby. While the system under test remains in modern standby, power down the Wi-Fi access point and wait a few minutes for the system to connect to the cellular network. After a few minutes, re-test push mail and Skype calls to validate that Windows automatically connected to the cellular network and re-established connectivity.

- **Test area:**Wi-Fi to MBB transitions during modern standby.
- **Purpose:**Ensure that the system automatically connects to Wi-Fi or MBB during modern standby.
- **System configuration:**
  - Factory image installed on the system.
  - All drivers loaded in Device Manager.
  - Factory Microsoft Store apps are installed.
  - System is running on battery power.
  - Wi-Fi and MBB are validated to function correctly with the screen on and manually configured connections to each network.

Test scenario	Expected result	Troubleshooting notes

Test scenario	Expected result	Troubleshooting notes
System can receive IM notifications during modern standby.	Ensure that all IMs received during modern standby are accompanied by sound notifications, while the display remains powered off. When the system wakes from modern standby, check that all IM messages have been received and that the badge on the lock screen is updated appropriately.	If a notification is not received, a trace is needed to diagnose the problem. See Capture and View a WPA Trace for Modern Standby Diagnosticsfor information on how to capture a trace and analyze the information.

Automated lab testing considerations

Article • 10/08/2021

System designers should consider automated lab testing scenarios when they design the power and battery charging subsystem. The most relevant consideration is platform recovery from crash or unresponsive system scenarios.

Platforms that are designed for automated lab testing, including platforms that are targeted for co-engineering programs with Microsoft, must implement configurable support for automatically booting the system when power is applied. Such automatic booting conflicts with the user experience goal that the platform remains off until the user physically turns it on; however, it is a requirement to achieve a fully automated test environment.

In Microsoft test labs, platforms are recovered from crash and unresponsive system scenarios by remotely removing and applying system power. The platform must be capable of ignoring the presence and charge capability of the internal battery and use only the presence of AC power to determine if the platform should automatically boot. **This behavior is for lab-enabled systems only and can be configured by using a UEFI variable that Windows sets.**

> **Note**
>
> Systems that are designed for users must not power on automatically when power is applied unless they are required to detect the power capability of the Universal Serial Bus (USB) charger.

UEFI DISABLEBATTERY variable definition

The DISABLEBATTERY variable is created under VendorGuid {0x77fa9abd, 0x0359, 0x4d32, {0xbd, 0x60, 0x28, 0xf4, 0xe7, 0x8f, 0x78, 0x4b}}. When the variable is present (that is, when GetVariable() returns EFI_SUCCESS), the requested platform behavior must be modified from the default battery behavior when the value is set.

When DISABLEBATTERY is set	When DISABLEBATTERY is not present or not set

When DISABLEBATTERY is set	When DISABLEBATTERY is not present or not set
• The platform must not use the presence of the battery to determine if the system should automatically boot or remain powered on. • If AC or external charger power is removed, the system should immediately power off. • If AC or external charger power is applied, the system should immediately power on.	• The platform should have behave according to the requirements for retail and/or end-user systems. • If AC or external charger power is removed, the system should remain on if sufficient power remains in the system battery. • If AC or external charger power is applied, the system should not automatically boot unless the system is capable of only USB-based charging and thus requires UEFI firmware or host software to determine the input power of the attached charger.

Modern Standby stress and long-duration testing

Article • 10/08/2021

System designers should run stress tests and long-duration tests on their Modern Standby systems to help identify and resolve potential reliability issues. Modern Standby enables the system to keep running, even when it is in a low-power, screen-off state. This state is different from the traditional ACPI Sleep (S3) and Hibernate (S4) states, in which much of the system hardware and software is stopped and then stays inactive until it is later restarted on resume.

Modern Standby enables the system to stay up and running for a much longer total time and can therefore expose hardware and software reliability issues that would not be discovered on a system that supports only S3 and S4.

Entry and exit

Every Modern Standby system should be validated to enter and exit Modern Standby for at least 1,000 cycles without failure. Entry to and exit from Modern Standby is the user's primary interaction with low-power operation on the system and should be extremely reliable.

Successfully entering and exiting Modern Standby validates a number of hardware, firmware, and device-driver components, which include:

- The platform hardware that manages power-button operation, including the power-management IC (PMIC).
- The display-panel management and initialization hardware.
- The Wi-Fi and networking device firmware and driver.
- The graphics device driver.

Stress-testing of Modern Standby entry and exit can be automated using the PwrTest tool. PwrTest should be installed on the target system as part of the Windows Driver Kit (WDK), which includes additional software for automating the system power button on Modern Standby systems.

Test scenario	Expected result	Diagnostic notes

Test scenario	Expected result	Diagnostic notes
The system can enter and exit Modern Standby reliably for at least 1,000 cycles.	Use the PwrTest tool and the /cs command-line option to automatically cycle the system through Modern Standby for 1,000 cycles. The expected result is that the system completes all 1,000 cycles.	We recommend incrementally increasing the stress test to 1,000 cycles. First, test for 100 cycles. If an error is found, connect the system to a kernel debugger and to the SoC hardware debugger, and repeat the 100-cycle test to capture and determine the root cause of the issue. After the 100-cycle test successfully completes, extend the cycle count to 500 cycles and then to 1,000 cycles.

SoC low-power state transitions

The firmware and drivers that are responsible for managing SoC transitions between the idle and active power states must be highly reliable to withstand the stresses of operating for long periods in Modern Standby. SoC low-power state transitions should be stressed through long-duration Modern Standby testing. This testing helps ensure that the system remains reliably operational during long Modern Standby durations, such as over the weekend. This test should be performed while connected to AC power.

Measurement scenario	Expected result	Power notes
The system can stay in Modern Standby for 100 consecutive hours and is functional on exit. The system maintains Wi-Fi connectivity during the 100 hours and Wi-Fi connectivity is functional on exit.	Put the system in Modern Standby and wake it with the power button after 100 hours. The expected result is that the system powers on instantly and the Wi-Fi connection is operational without additional configuration or selection of a Wi-Fi network.	We recommend incrementally increasing the long-duration test to 100 hours. First, test for 24 hours. If an error is found, connect the system to a kernel debugger and to the SoC hardware debugger, and repeat the 24-hour test to capture and determine the root cause of the issue. After the 24-hour test successfully completes, extend the duration to 100 hours.

Windows HLK Modern Standby stress test

The Windows Hardware Lab Kit (HLK) includes a Modern Standby stress test named Connected Standby Stress with Driver Verifier's Concurrency Stress that exercises automatic modern standby transitions at the same time that device drivers are exercised for device operation. The test is designed to verify that the device and its driver(s) continue to function as the system transitions to and from the Modern Standby power state.

This test is a critical part of validating that the system continues to operate as expected after it exits Modern Standby. This test is included as part of the Windows HLK and is required for system certification.

Test operation

The test uses the Windows Device Testing Framework (WDTF) SimpleIO interfaces to exercise devices that are enumerated on the system. These devices include sensors, cameras, audio, graphics, Wi-Fi, storage, and Bluetooth devices. The test places the system in Modern Standby for one minute, and then transitions the system out of Modern Standby and exercises the devices for 30 seconds. This cycle repeats 150 times.

During test execution, Driver Verifier is enabled to help identify driver bugs and memory leaks.

The test helps identify the following system or device driver problems:

- A system becomes unresponsive or crashes during device operation after a Modern Standby session.
- The inability for the system to enter the low-power state (deepest runtime idle platform state, or DRIPS) after device activity.
- Driver issues that are identified by Driver Verifier, including system corruption, driver failures, and memory leaks.
- Driver issues after resume from Modern Standby, including unresponsiveness, crashes, or problem codes.

Resolving test failures

The test exercises multiple devices, which can result in different types of test failures. Identifying the type of test failure is the first step to finding the root cause of system or driver issues.

The test typically fails in one of the following three failure modes:

1. The test fails and the failure is recorded in the Windows HLK logs, which contain data about the detected failure.
2. The test fails, but the system does not report to the Windows HLK server as a result of the failure; however, the system is responsive and works with local interaction.
3. The test does not complete and the system under test crashes or becomes unresponsive (frozen at a black screen).

Debugging test failures that are recorded in the Windows HLK logs

There are two common failure types when test failures are recorded in the Windows HLK logs:

- The system failed to enter the low-power state (DRIPS) during the test.
- The test detected that it could no longer communicate with a driver, and a time-out occurred.

You can use the SleepStudy report, which is included as part of the test logs, to identify which components are responsible for preventing the system from entering the low-power state (DRIPS). There are several common causes:

- Test setup and configuration problems, including using a wired Ethernet adapter that does not support NDIS 6.3 and Modern Standby functionality.
- DHCP server problems on the wired LAN network.
- A device and/or driver that does not correctly idle to its own low-power mode during Modern Standby.

The test logs might also include a failure message that indicates which devices did not respond to I/O requests in a timely manner. This condition is considered a test failure because it can prevent the user or an app from being functional when the system resumes from Modern Standby.

The test logs indicate the last devices to perform I/O operations—these devices are the source of the test failure. The test log output in the following example shows that the ACPI\XXXX\2&DAFA3FF&1 device timed out.

Message	7/16/2013 12:50:24.333 AM	WDTF_SIMPLEIO_STRESS_PROC : - WaitAsyncCompletion(Some Location Sensor Device ACPI\XXXX\2&DAFA3FF&1)
Message	7/16/2013 12:59:50.333 AM	WDTF_SIMPLEIO_STRESS_PROC : - WaitAsyncCompletion(Some Other Device XXX_XXX\UART_XXX\3&2F829BAD&0&F00D)

A common cause of failures is poor GPS reception, which causes the GPS device to take extremely long amounts of time to reply to I/O requests. For more information about running this test on systems with GPS devices, see Notes for systems that are equipped with GPS.

Debugging test failures without logs (and a responsive system)

If the system under test is still running with no signs that the test is still running, the most likely cause is that the system has encountered a fatal error or restarted. To debug these issues, check the system directory for any dump files, and disable any hardware watchdog that might reset the system.

Debugging test failures when the system is unresponsive (black screen)

If the system is frozen on a black screen, a kernel debugger must be must connected to the system to diagnose the problem.

If the kernel debugger is already connected and the system is not responding to the kernel debugger, a hardware debugger is required to identify the reason that the system locks up. You can consult with the core silicon/SoC provider for additional assistance with debugging.

Additional HLK Documentation

- HLK Test Reference
- Tools Available

Notes for systems equipped with GPS

If the system-under-test has a GPS device or location sensor device, the following Windows settings must be enabled before running the test:

- **Control Panel\Hardware and Sound\Location Settings\Turn on the Windows Location platform**
- **PC Settings\Privacy\Location: Let Windows and apps use my location**

You can use the Sensor Diagnostic Tool in the Windows Driver Kit (WDK) to confirm the reception of the GPS signal at the test site. For more information, see Testing sensor

functionality with the Sensor Diagnostic Tool.

Modern Standby resume performance

Article • 11/05/2020

For users, the most noticeable benefit of using a Modern Standby PC is its ability to instantly resume from sleep. Modern Standby resume performance is typically less than one second, as measured from power button press to powering on the display. Significant engineering in Windows, third-party drivers, and system firmware is required to achieve this high level of performance.

You can measure Modern Standby resume performance by using tests that are built into the Windows Hardware Lab Kit (Windows HLK) or by manually capturing a trace and reviewing the result in the Windows Performance Analyzer (WPA).

Watch this video to learn how to use WPA to measure the time required to wake the platform from Modern Standby.
https://learn-video.azurefd.net/vod/player?id=72e81555-45c1-4106-a47a-25a606a64e33&locale=en-us&embedUrl=%2Fwindows-hardware%2Fdesign%2Fdevice-experiences%2Fmodern-standby-resume-performance

Using WPA to measure Modern Standby performance

To evaluate connected standby resume performance, take a trace with the provider named Microsoft-Windows-PDC, which is the system-supplied power dependency coordinator. The `Trace_start.cmd` file that is included with the WPA package includes this and other providers that are required to measure resume performance. The WPA package is available in the Windows Assessment and Deployment Kit (Windows ADK) download and includes scripts and documentation for modern standby analysis.

To collect a trace

1. Open a Command Prompt window as Administrator.
2. Run the Install.cmd command to install the package.
3. Follow the instructions in Capture a WPA trace to collect the trace. For validating resume performance, a 5-10 minute session should be sufficient.
4. To save the trace file, rename it or save it to a different location because a file with this name will be overwritten the next time a trace is taken.
5. Repeat steps 3-4 to gather another trace.

After you have completed these steps, open the trace file in WPA by double-clicking it. (If this doesn't work, make sure you are using Wpa.exe to open the trace file.)

Validate that the system entered the lowest power state (HW DRIPS) during the modern standby session. Resume performance should be measured for transitions from HW DRIPS to the screen turning back on because these transitions have the longest latencies.

Check the DRIPS graph under the power graphs to determine whether the system successfully entered DRIPS while in modern standby.

If the system did not enter DRIPS, the exit latency value is not used in the Windows HLK test and the Windows HLK test will fail. In this case, you should rerun the WHCP test.

If this issue occurs consistently, stop the analysis of resume performance, and identify and resolve why the system is not entering DRIPS.

Using WPA to optimize Modern Standby performance

> **Note**
>
> The two most common causes of modern standby exit delay are turning on the display and network devices—in particular, mobile broadband (MBB) radios.

The Windows power dependency coordinator (PDC) transitions through a number of phases for each entry to and resume from modern standby. These PDC phases always take place in series for entry and in reverse order for resume. The duration in time of each phase can vary.

You can use WPA to take a trace to indicate where most of the time is spent during resume from Modern Standby. To analyze where the Modern Standby exit time is being spent, zoom into just the region between the start and end of the exit from Modern Standby. Pull up the **PDC Notification Phase** graph under **Power**. This graph lists each connected standby PDC phase.

Highlight all the phases shown in this graph to get the total time spent in the PDC phases.

You can also get the duration of a particular PDC phase. To do so, change the display format of the **PDC Notification Phase** graph to **Both graph and table**. Then, expand the phase of interest and look for the duration.

Common causes of long modern standby resume

Some common problems might cause long latencies during resume from modern standby. The most likely causes of such latencies are turning on the display and re-enabling network connections.

The Windows Performance Analyzer (WPA) is used to obtain the graphs that are discussed in this section. For more information, see Using Windows Performance Analyzer to analyze modern standby issues.

Display issues

From the time that the system completes the PDC phases to the end of modern standby exit, a variety of processes can cause delays. The foremost cause of delays is turning on the display. The request to turn on the display is sent at the end of the PDC phases while exiting modern standby.

To determine the time required to turn the display on:

1. Find the event time stamp for the initial request to turn on the monitor.

   Navigate to the **PDC Notification Phase** graph > **Connected Sessions Check** event > **Exit Time**.

2. Find the event time stamp for when the kernel received the notification that the monitor finished turning on.

   Navigate to the **Generic Events** graph > **Microsoft-Windows-Kernel-Power** provider > **Task Name** column > **DisplaySessionStatus** event.

3. Take the difference of these two event times, and the result is the time that is required to turn on the display.

If this value is greater than expected, contact the system graphics device vendor.

Network connectivity issues

Re-enabling network connectivity is a major cause of long modern standby resume latency. The time required to finish the resiliency phase (so that the next PDC phase can start) depends on how quickly the D0 IRP for the required network device can be completed. (A D0 IRP is an IRP_MJ_POWER IRP that specifies a DEVICE_POWER_STATE enumeration value of PowerDeviceD0.)

If a Wi-Fi connection is available, the system will wait for the Wi-Fi device only, regardless of whether a mobile broadband (MBB) connection is available. Otherwise, the system will wait for the MBB device to complete the D0 IRP.

If no network device is available, the system does not wait for these devices before proceeding to the next phase. As a result, the time spent in the resiliency phase often reflects the time that is needed to complete the D0 IRP for the network device.

To find the time taken to complete the D0 IRP:

1. Pull up the **Device Dstate** graph under **Power**.
2. Zoom into the section for **Connected Standby Exit**.
3. Look for devices that are changing D-states (device power states) during this time. For a system that has Wi-Fi enabled, look for the Wi-Fi device.
4. Find the column labeled **D-Irp Duration (ms)**. (Look for this exact column name—there are other columns with similar names.) This number indicates how long it took to complete the IRP for the D-state transition.

With mobile broadband, the system takes about twice as long to resume from modern standby.

If mobile broadband is enabled on the system, try removing the SIM card and retrying the modern standby exit test. The Windows HLK test only checks that Wi-Fi is connected.

If the latency of any network device seems long, contact the Wi-Fi or mobile broadband device vendor for the system.

Modern standby SleepStudy

Article • 06/21/2022

Starting with Windows 8.1, a software tool, SleepStudy, became available as an inbox component in all Windows PCs that implement the modern standby power model. SleepStudy can measure modern standby performance with minimal impact. Details on the report's contents can be found below.
https://learn-video.azurefd.net/vod/player?id=fb7b986a-d2cd-462c-a9ec-9fb946050baf&locale=en-us&embedUrl=%2Fwindows-hardware%2Fdesign%2Fdevice-experiences%2Fmodern-standby-sleepstudy

Watch this video to learn how to use SleepStudy to find and fix components that cause unexpected battery drain.

Tracking system activity and battery drain during modern standby can be difficult because the tracking itself can cause unnecessary activity and battery drain. For example, traditional disk-based logging has the unwanted side effect of causing excessive battery usage when the disk is activated for logging. In contrast, the SleepStudy tool is designed to avoid generating activity that could interfere with the modern standby performance that it measures.

The most detailed way to measure power consumption during modern standby is to use an instrumented system, which is a physical system that has power measurement leads connected to every major hardware subsystem of interest. However, testing to this level of detail is not practical in many cases because of engineering cost, and systems that have already been sold to customers typically cannot be tested in this way.

The SleepStudy tool provides overview information about each modern standby session. This information includes the active time, the idle time, and the power consumed. A session starts when the system enters the modern standby state, and ends when it exits this state.

SleepStudy also provides first-level information about the causes of activities that occur during each modern standby session. This feature allows for easy investigation of long-running activities.

Battery Information: Each SleepStudy report concludes with information about the system battery configuration. In addition to name and manufacturer, this information includes battery size and design capacity. The battery size and design capacity are particularly important for SleepStudy because they are taken into account when estimating modern standby battery life.

In some cases, the capacity ratio will exceed 100 percent. This is expected and will change over time as the battery chemistry changes.

For more information about running SleepStudy and interpreting the results, please reference the following videos:

- Channel 9 video: Powercfg Sleep Study
- Channel 9 video: System Power Report

Running SleepStudy

The SleepStudy tool runs from a Command Prompt window and is simple to use. SleepStudy outputs an easy-to-read HTML report.

To run SleepStudy, open a Command Prompt window as Administrator and enter the following command:

```
powercfg.exe /SleepStudy
```

In response to this command, the built-in powercfg.exe command-line tool creates an HTML file named Sleepstudy-report.html in the current working directory.

Advanced Options

By default, the SleepStudy report covers the last three days of system operation. To change the duration covered by the SleepStudy report, use the powercfg.exe tool's /duration option. With this option, you specify an additional parameter, which is the number of days (up to 28) that the SleepStudy report covers.

For example, to generate a SleepStudy report for the last seven days of system operation, open a Command Prompt window as Administrator and enter the following command:

Windows Command Prompt

```
powercfg /sleepstudy /duration 7
```

For more information about powercfg.exe, see Powercfg command-line options.

SleepStudy Reports supporting TShell

On Windows versions after 2004, SleepStudy reports can be generated directly as HTML on mobile images supporting TShell. To do so, connect to TShell, navigate to writeable directory in data partition (e.g., 'cd c:\data\') and run 'powercfg /sleepstudy'. On Windows versions 2004 and earlier, the following steps are required after navigating to the writeable directory to generate the report as XML and convert it to HTML.

1. Run the following command:

   Windows Command Prompt

   ```
   powercfg /sleepstudy /xml
   ```

2. Copy **sleepstudy-report.xml** that is generated in the local directory to the host machine.

3. Run the following command:

   Windows Command Prompt

   ```
   powercfg /sleepstudy /transformxml <path to sleepstudy-report.xml>
   ```

Sleep study traces are automatically generated on mobile for screen off sessions longer than 10 minutes and are maintained for the last 7 days. The reports contain the last 3 days, by default, but can be extended with the '/duration' flag.

Report Details

Beginning with Windows version 2004 on Modern Standby systems, the SleepStudy report is organized as a series of state changes -- primarily Active, Screen Off, and Sleep states. Therefore, while an overall modern standby session is defined as one instance of the screen turning off and back on again, the SleepStudy report will break this down into the time spent quiescing the system (the *Screen Off* state), and the time spent in its long term low power state (the *Sleep* state). Prior releases will just show details for the overall Modern Standby session.

For each state, an overview is provided at the top of the report with a hyperlink to its detailed section in the HTML report.

The SleepStudy report also contains:

- Static configuration information about the hardware platform, its operating system installation, and firmware version.
- A graphical view of the usage trend over the last 72 hours.
- A summary table of each state segment, which includes:
  - Start time, end time, and duration.
  - Power source (AC or battery power).
  - Battery power consumed and average power consumption.
- Each *Screen Off* session includes additional detailed information:
  - The information contained in the summary table.
  - Any blockers that prevented the system from quiescing immediately to enter sleep.
- Each *Sleep* session includes additional detailed information:
  - The information contained in the summary table
  - The five most active components ("top offenders") in the modern standby session. The information shown for each component includes the component type, name, and device path (if applicable).
  - A histogram chart of idle time.
- Information about the system battery configuration, including design capacity and cycle count.

The remainder of this topic describes a sample SleepStudy report, and explains how to interpret the report and understand the information that it contains.

System information

Every SleepStudy report begins with basic system information, which includes system name and firmware version. This information is essential because changes to the operating system, firmware, and BIOS can have significant impact on modern standby battery life.

Usage trends

Every SleepStudy report includes a graphical view of system usage that includes modern standby periods.

The graph is color-coded. Green, orange, and red segments correspond to low, medium, and high system activity, respectively.

The default period that the graph covers is three days.

The graph consists of dotted, solid, and no-line segments, which indicate AC power, battery power, and system-off periods, respectively. The following graph legend is

included with each SleepStudy report.

Summary information

Each SleepStudy report includes a table that summarizes the modern standby sessions that are evaluated in the report.

Each row in the summary table contains information about one state of a modern standby session. The rows in the summary table are color-coded to identify sessions that could be investigated for improvement.

Color coding

Beginning in Windows version 2004 with the seaparation of Modern Standby sessions into *Screen Off* and *Sleep* states, colors for a row in sleep study reports are based on a few main rules:

1. If the session is less than 2 minutes, no detailed data is shown.

    - If it's a Screen Off session, then it is colored Green, since most screen off sessions are expected to be very short.
    - If it's a Sleep session, then it is colored Grey, since there is no information to determine if it's a good or bad session based on drain or time spent in DRIPS.

2. If the session is 2 minutes or greater, then both *Sleep* and *Screen off* sessions will be colored according to the following drain and DRIPS % thresholds.

    - For Screen Off sessions, this is:
        - Red indicates that the drain rate is >= 1% per hour
        - Yellow indicates that the drain rate is between .33 and 1% per hour
        - Green indicates all other cases
    - For Sleep sessions, this is:
        - Red indicates that the DRIPS rate is <80% or the drain rate is >= 1% per hour
        - Yellow indicates that the DRIPS rate is between 80-94% or the drain rate is between .33-1% per hour
        - Green indicates all other cases

3. If there is a red child blocker, then the parent session will be colored red as well. Clearly incorrect third party child blockers will be colored purple.

By design, some sessions are active (red rows). But most sessions can be expected to reflect low activity and low power (green). The color scheme is designed to make

potential high-drain sessions easy to identify. You should be most concerned about long sessions (of several hours) that show high sustained battery drain—these sessions have the most impact on the battery.

Legacy color coding

Prior to 20H1, the color coding for each overall modern standby session is based on a combination of the drain rate and the deepest runtime idle platform state (DRIPS) rate (percent low power). The color is determined by the worst performance of either drain rate or DRIPS rate, according to the following rules:

- Red indicates at least one of the following:
  - The DRIPS rate is < 80 percent.
  - The drain rate is >= 1 percent per hour. (If the drain rate is >= 1 percent per hour, the modern standby battery life will be at most four days.)
- Orange indicates at least one of the following:
  - The DRIPS rate is between 80 and 94 percent.
  - The drain rate is between 0.333 and 1 percent. (If the drain rate is under 0.333 percent, the platform will achieve 12+ days battery life in modern standby.)
- Green indicates all other cases.

Drain rate calculation

SleepStudy calculates drain rate by using remaining capacity information that is provided by the platform battery and charging subsystem. The battery capacity, in milliwatt-hours, is recorded at the start and end of each SleepStudy session. The battery capacity information is provided by the platform through the ACPI _BST control method under the battery device object in the ACPI namespace.

List of summary information

The summary table includes the following basic information (from left to right):

- The session number (left column) starts with one, and increments for each session that is reported during this period. The default report period covers the last three days.
- START TIME (local time) is shown in YYYY-MM-DD HH:MM:SS format. Additional sessions on the same day do not repeat the year-month-day information.
- DURATION, in hours:minutes:seconds, of the modern standby session. This duration time covers the approximate period between the transition to screen off and subsequent transition to screen on.

- ENERGY CHANGE shows the number of absolute milliwatt-hours (mWh) that are consumed and the relative percentage of the battery's last full-charge capacity. A session in which no change occurs in remaining capacity is indicated by a hyphen (-), as shown in session 6 in the summary table in the preceding example.
- ENERGY CHANGE shows the number of absolute milliwatt-hours (mWh) that are consumed and the relative percentage of the battery's last full-charge capacity. A session in which no change occurs in remaining capacity is indicated by a hyphen (-), as shown in session 6 in the summary table in the preceding example.
- CHANGE RATE, in milliwatts, and AC (Charge) or DC (Drain) power source indicator. The change rate is calculated by dividing the ENERGY CHANGE value by the DURATION value.
- % LOW POWER STATE TIME shows the DRIPS rate and (if applicable) hardware DRIPS as the percentage of time in which the SoC resides in the lowest power state (DRIPS). Hardware DRIPS (denoted by HW: preceding the residency percentage) is available on Intel and Qualcomm SoC-based Windows PCs only.

Overall modern standby sessions of less than ten minutes are not tracked by the SleepStudy report. To measure modern standby performance, overall session durations should be greater than ten minutes. Longer periods (more than one hour) are reflective of real user experience.

Energy change is not tracked for AC sessions (indicated by the Charge keyword under CHANGE RATE). This information is captured in other reports, such as the report that the powercfg/energy command generates, because energy policies that are used when the platform runs on AC power are different from the policies that are used when the platform runs on battery (DC) power. In general, these policies are less strict and allow for different use cases. Every row in the summary table is a hyperlink to per-session details that are presented in the SleepStudy report.

Session details

A per-session details section of the report starts by repeating the session information from the summary table.

The table can show the following types of offenders:

- Fx Device. A device that has a driver that implements support for the Windows power framework (PoFx). Typically, this type of device resides on the SoC itself.
- Activator. A software component that can keep the system active to perform valuable work during modern sleep. (In the preceding example, the name BI identifies the broker infrastructure, which is a Windows software component that coordinates background task execution.)

- Networking. A networking subsystem device or component.
- Processor. CPU active time that occurs outside of an activator being enabled.
- PDC Phase. Time that is spent in the different phases of entering or exiting modern standby-- this is most applicable to *Screen Off* states. Please see Prepare software for modern standby for more information.
- Other. Includes miscellaneous tracking information. For example, CPU utilization outside of an enabled activator is listed under TYPE as Other.

Each component in the Top Offenders table is color-coded by activity time. If it is over ten percent busy, the row is highlighted in red. If the component is between five and ten percent active, it is colored orange. Otherwise, the component row is highlighted in green.

Exit reasons

The detailed info section also includes exit reason for Sleep and Screen off sessions.

Sleep and screen off exit reasons

The below exit reasons are reasons that the system can wake from Modern Standby (Sleep or Screen off) to Active.

Expand table

Exit reason code	Exit reason
0	Unknown
1	Power Button
3	SC_MONITORPOWER
4	User Input
5	AC/DC Display Burst
6	User Display Burst
7	PoSetSystemState
8	SetThreadExecutionState
10	Session Unlock
11	Screen Off Request

Exit reason code	Exit reason
12	Video Idle Timeout (VIDEOIDLE)
13	Policy Change
14	Sleep Button
15	Lid
16	Battery Count Change
17	Grace Period
19	Dynamic Partitioning
20	Hibernate, or Shutdown
21	System Idle Timeout (STANDBYIDLE)
22	Proximity Sensor
23	Thermal Standby
25	Resume S4 Display Burst4
26	Terminal
27	PDC Signal
28	AC/DC Display Burst Suppressed
30	Winrt API
31	Input Keyboard
32	Input Mouse
33	Input Touchpad
34	Input Pen
35	Input Accelerometer
36	Input Hid
37	Input UserPresent
38	Input SessionSwitch
39	Input Initialization
40	PDC Signal: Windows Mobile Power Notification

Exit reason code	Exit reason
41	PDC Signal: Windows Mobile Shell
42	PDC Signal: Hey Cortana
43	PDC Signal: Holographic Shell
44	PDC Signal: Windows Biometric Framework Fingerprint
45	Directed DRIPS: Device-S4
46	Monitor Dim
47	Builtin Panel
48	Display Required Undim
49	Battery Count Change Suppressed
50	Transition From Sleep
51	Terminal Initialization
52	PDC Signal: Sensor - Human Presence Detected
53	Battery PreCritical
54	Input Touch
55	Restricted Standby Battery Drain Budget Exceeded
56	Restricted Standby Doze Timeout
57	Smart Restricted Standby

Sleep only (not Screen off) exit reasons

The below exit reasons are reasons the system can wake from Sleep to Screen off, without entering an active state.

Expand table

Exit reason code	Exit reason
16777216	PDC Task Client: Unknown
16777217	PDC Task Client: Network Refresh
16777220	PDC Task Client: Maintenance Scheduler

Exit reason code	Exit reason
16777221	PDC Task Client: Sync Client
16777222	PDC Task Client: Tethering Client
16777223	PDC Task Client: SleepStudy Accounting
16777224	PDC Task Client: Windows Update Client
16777225	PDC Task Client: Wake on LAN
16777227	PDC Task Client: Terminal Server Remote Session
16777228	PDC Task Client: Servicing Session
16777229	PDC Task Client: User Mode Shutdown Client

See also

- Powercfg Sleep Study
- System Power Report

Modern standby SleepStudy common problem examples

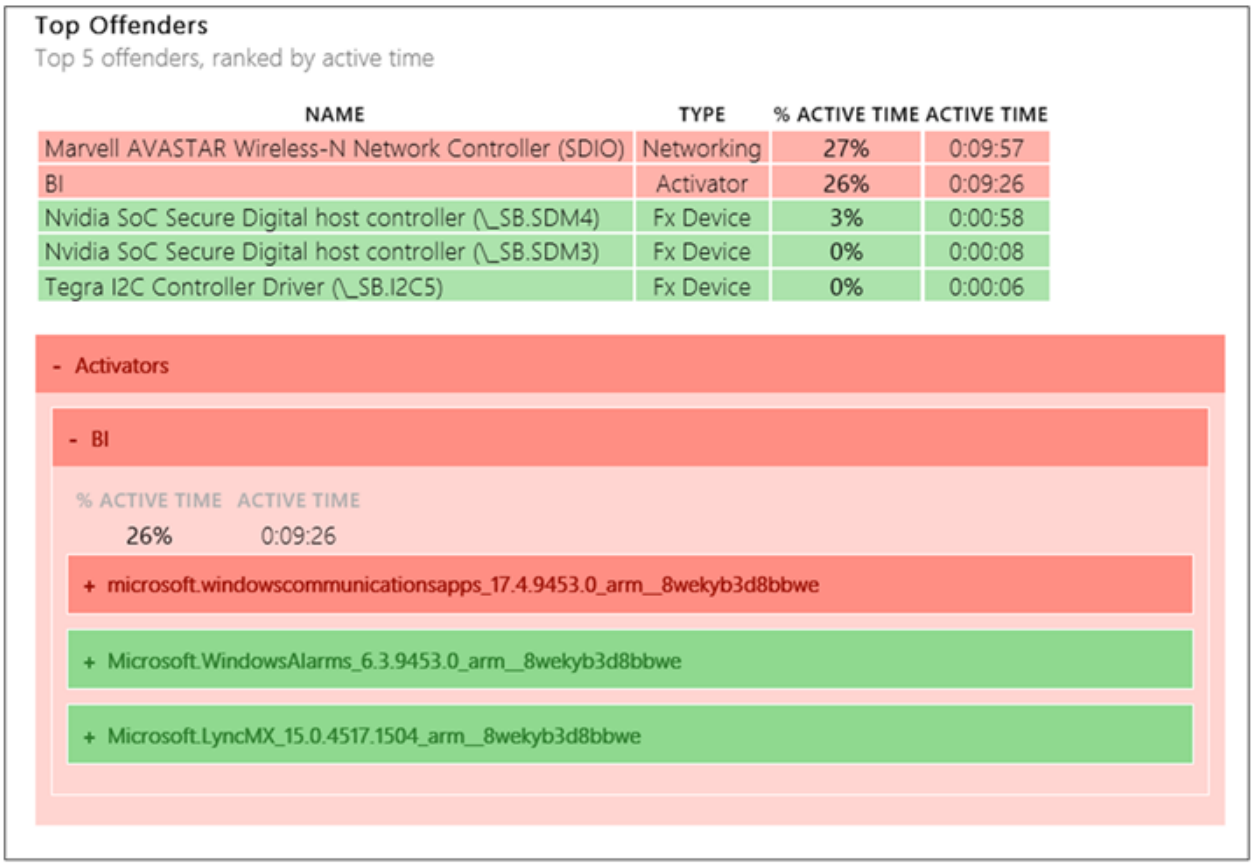
Article • 01/03/2020

You can use the SleepStudy report to observe and diagnose common modern standby problems. The examples in this article show how to use the report to investigate problems such as high email or network activity, high battery drain, and inability to enter DRIPS.

High email activity

This example shows how software can keep the wireless LAN (WLAN) device up. The cascading effect of email through the Windows communication app keeps the broker infrastructure (BI) system active. BI, in turn, keeps the WLAN network up so that the system stays up-to-date with emails.

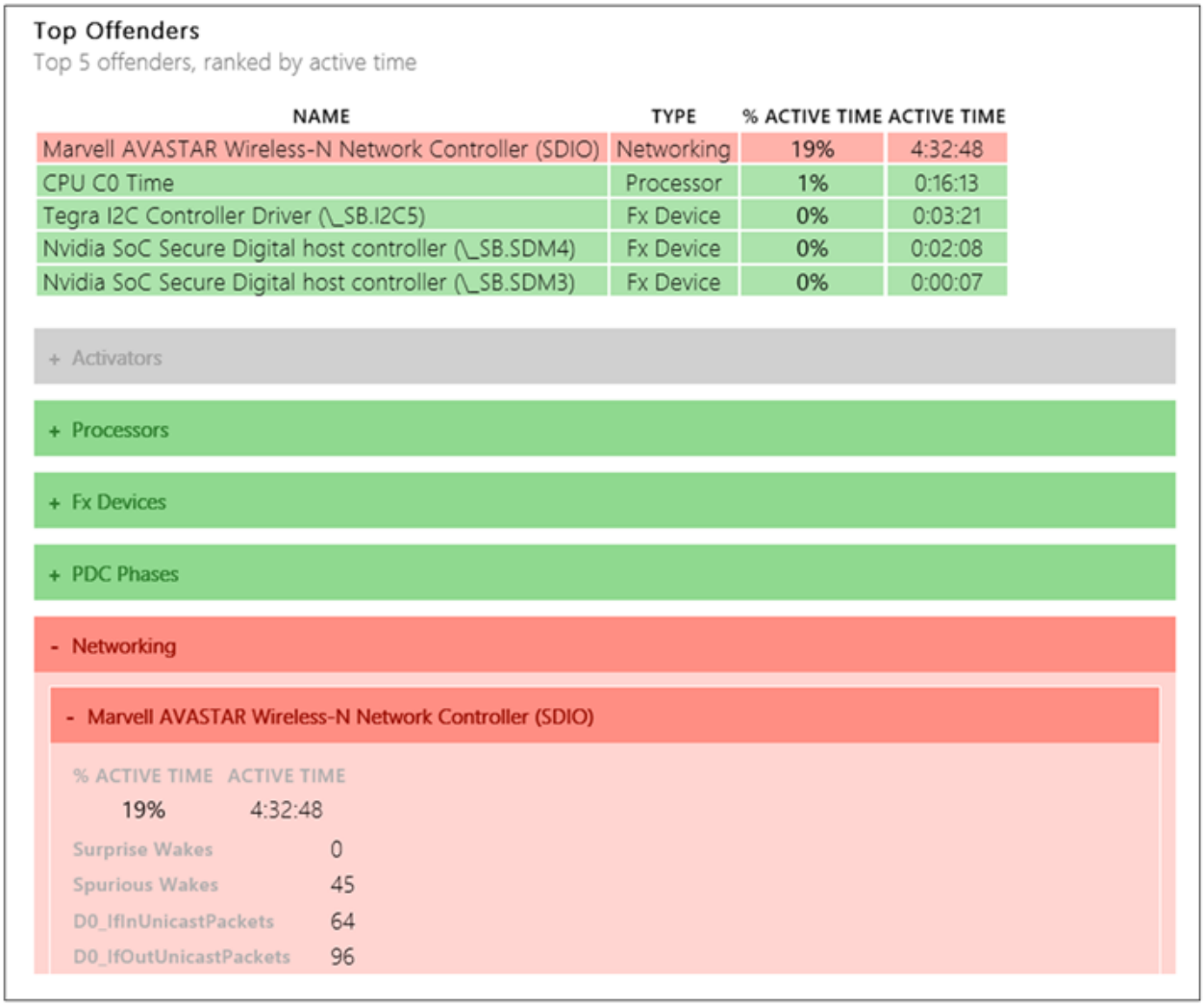
In the following image, the SleepStudy report shows the relationships between the software and hardware components and how they collectively affect the battery.



High network activity

This example shows how external events, such as WLAN environment, can keep the WLAN device up. The WLAN device might have a challenging radio environment and the Windows system might not be able to establish a reliable Internet connection. We see how these events affect the WLAN device, which, in turn, impacts the battery.

As shown in the following image, SleepStudy captures this device activity as a percentage of the modern standby session and warns of any durations that are over ten percent.

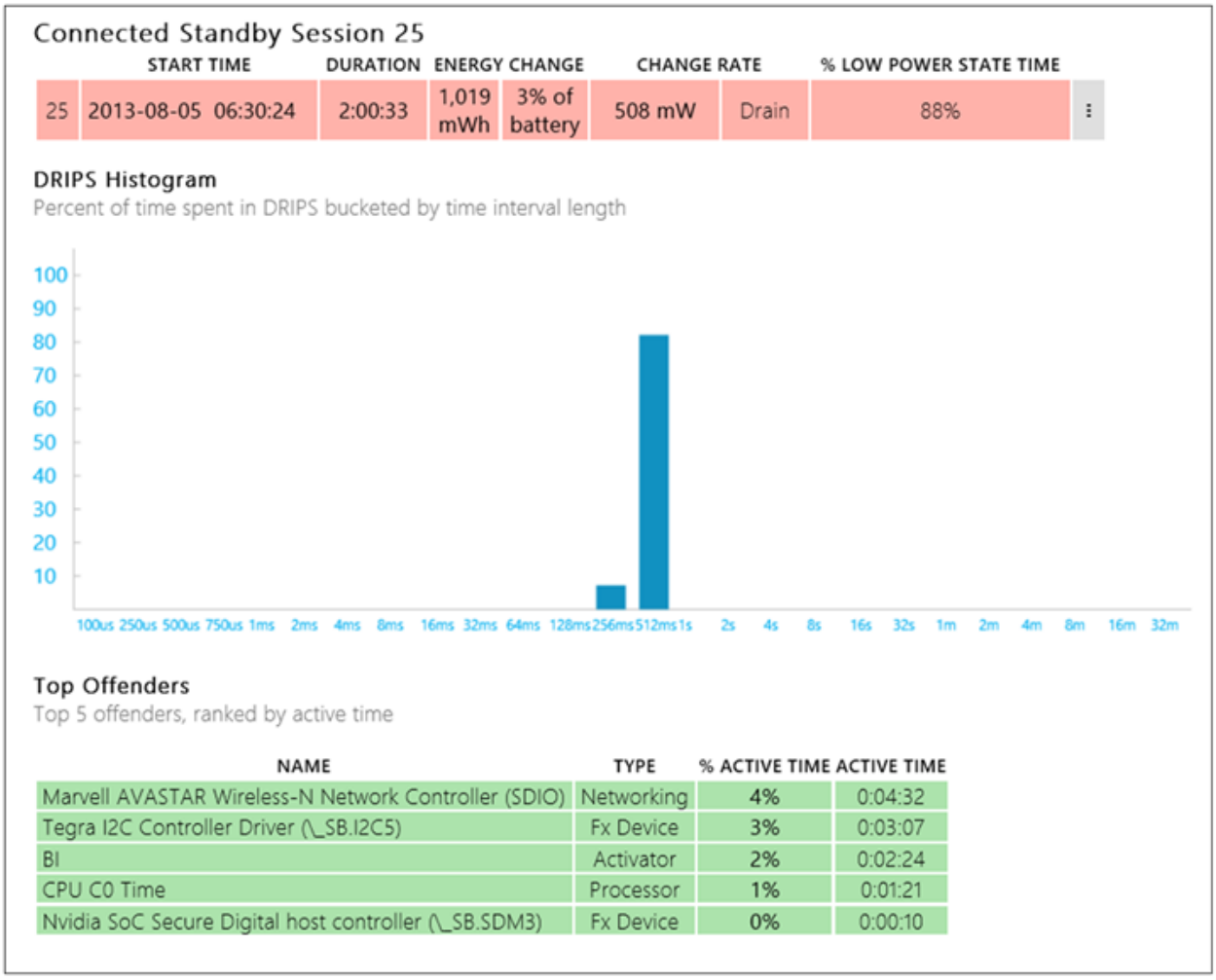


High drain and obvious offender

In some cases, the SleepStudy report shows high battery drain but does not identify an obvious offender that is highlighted in red.

The example in the following image shows that the modern standby session had very high drain, but that no offender was over ten percent active. The deepest runtime idle platform state (DRIPS) histogram shows wake-up patterns. Typical low-power sessions have sleep time intervals that are close to 32 seconds, but this histogram shows that the sleep time interval is less than one second (closer to 512 milliseconds). This type of

activity implies that frequent interrupts or some other hardware component causes the system to stay awake, which results in poor battery life.



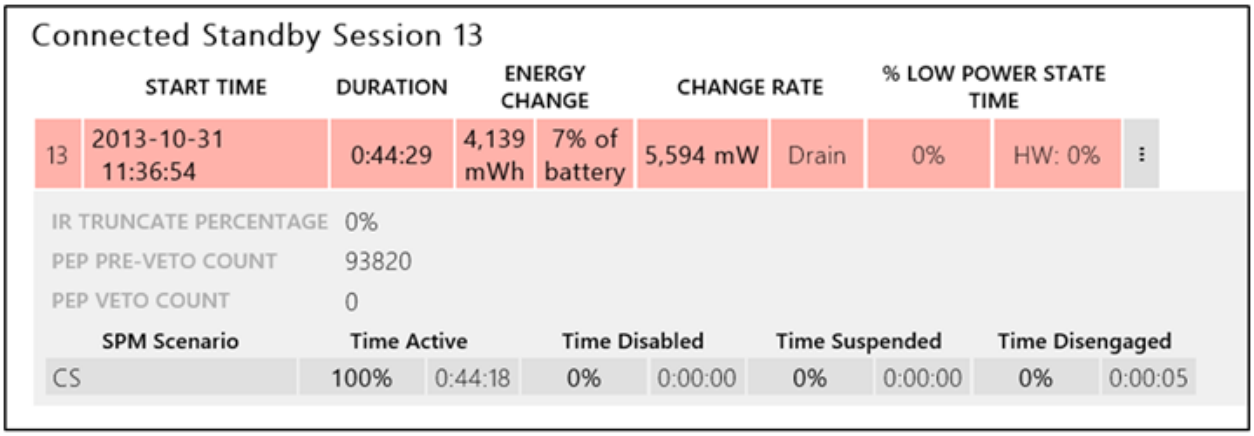
For this example, a you can view a trace to identify the component that is causing the system to frequently exit the low-power state (DRIPS). For more information, see Capture and View a WPA Trace for Modern Standby Diagnostics.

No software DRIPS or hardware DRIPS due to a missing driver

Some core silicon or System on a Chip (SoC) designs report both software DRIPS and hardware DRIPS. Software DRIPS refers to the Windows power manager's perception of when the SoC can be placed into the low-power state. It is based on device power state and CPU idle time.

Hardware DRIPS refers to the actual physical residency of the SoC in its lowest power state, as controlled by the on-SoC controller or microcode. Some SoC designs have an embedded controller or microcode that is responsible for actually transitioning the SoC to the lowest power state. The hardware might delay or prevent entry to the low-power state for reasons that are not perceptible to Windows.

A common reason for a modern standby session to have zero percent software and hardware DRIPS is that a critical driver is not loaded on the system. The summary table for the example modern standby session reports zero percent low-power state time for both software and hardware. When the summary details are expanded, as shown in the following screenshot, the value for PEP PRE-VETO COUNT is extremely high (it should nominally be zero). In this scenario, look for missing drivers in Device Manager.



Using Windows Performance Analyzer to analyze Modern Standby issues

Article • 01/08/2021

The Windows Performance Analyzer (WPA) displays traces of system activity in a graphical format. WPA is used for many Windows performance and debugging scenarios, and is the second-level triage tool for Modern Standby issues that cannot be resolved by using SleepStudy. WPA presents a graphical format of a trace file that contains events collected during a Modern Standby session.
https://learn-video.azurefd.net/vod/player?id=82566435-8b0e-4ad6-831e-2963bfdc5f13&locale=en-us&embedUrl=%2Fwindows-hardware%2Fdesign%2Fdevice-experiences%2Fusing-windows-performance-analyzer-to-analyze-modern-standby-issues

Watch this video to learn how to use WPA to analyze traces of Modern Standby sessions.
https://learn-video.azurefd.net/vod/player?id=0e45773f-2fb0-476e-8638-6bf77f892b92&locale=en-us&embedUrl=%2Fwindows-hardware%2Fdesign%2Fdevice-experiences%2Fusing-windows-performance-analyzer-to-analyze-modern-standby-issues

This video shows how to use the Platform Idle State graph and PDC Resiliency Activity graph to identify the cause of software activity that prevents the hardware platform from spending sufficient time in the DRIPS state.
https://learn-video.azurefd.net/vod/player?id=0e45773f-2fb0-476e-8638-6bf77f892b92&locale=en-us&embedUrl=%2Fwindows-hardware%2Fdesign%2Fdevice-experiences%2Fusing-windows-performance-analyzer-to-analyze-modern-standby-issues

Watch this video to learn how to use the Platform Idle State graph and Device Dstate graph to track down a hardware device that causes the hardware platform to spend too little time in the DRIPS state.

For more information about the Platform Idle State graph, see the "Common WPA graphs for connected standby power management" section below. For more information about the PDC Resiliency Activity graph and Device Dstate graph, see the "View a WPA trace" section below.

WPA is available in the Windows Assessment and Deployment Kit (Windows ADK) download package and includes scripts and documentation for Modern Standby analysis.

The remainder of this section refers to the documents and scripts that are provided in this download.

Capture and view a WPA trace for Modern Standby diagnostics

Trace capture is the key diagnostic method that is used to debug issues that are observed during Modern Standby through SleepStudy or other tools. A trace contains detailed information about system platform states, device states, software activity, CPU utilization, memory utilization, and other system events. Events that are captured in a trace show exactly what happened during Modern Standby and any problems that resulted.

Capture a WPA trace

Capture a trace of at least one hour of Modern Standby to observe trends and averages.

Use the following method to capture a WPA trace using the Windows Performance Recorder (WPR) with the Power profile:

1. Install the Windows Performance Toolkit (WPT).
2. Open an elevated command prompt and navigate to the WPT install location.
3. To start the trace, run: `wpr -start Power`
4. While recording, put the system into Modern Standby. Wait for at least one hour, then wake up the system.
5. To stop and save the trace into an event trace log (ETL), run: `wpr -stop <filename>.etl`

View a WPA trace

Use the WPA tool to view and analyze Modern Standby traces. Download the WPA tool, install it on a computer, and follow these instructions to open the trace file:

1. Run Wpa.exe. Note that Wpa.exe is available for x86 and x64 only.
2. In the WPA menu, click File, click Open, and select a trace file.
3. To apply a profile, click Profiles\Apply to open a separate Analysis tab.
4. Click Browse and select the applicable profile to apply.
5. Add other graphs to the current analysis view from Graph Explorer by following these steps:
   a. Expand a graph category in Graph Explorer.

b. Select the graph to add and drag it to the Analysis View pane.

To correlate data from a SleepStudy report to the WPA trace, use the mapping shown in the following table.

Expand table

SleepStudy	WPA trace
Activators	The **PDC Resiliency Activity** graph shows a list of activators that were active during the Modern Standby session.
Processors	The **CPU Idle States** graph shows a list of the CPUs in the system and their respective states.
Fx devices	The **Device Dstate** graph shows the list of Windows power framework (PoFx) devices that were active during the Modern Standby session.
PDC phases	The **PDC Notification Phase** graph shows the details of all the PDC phases.
Networking	Several graphs show networking activities. The **PDC Resiliency Activity** graph shows activators such as the broker infrastructure (BI) or Windows Push Notification Services (WNS) that can trigger network activities. The **Device Dstate** graph shows information about the activity of the Wi-Fi device. The **Generic Events** graph can show events that are triggered by networking components such as WCM, DHCP, and TCPIP.
Power Requests	The **Power Requests** graph shows details for all power requests that were active during this session. The relevant request types for Modern Standby are "System Required" and "Execution Required" power requests. "Display Required" are used for screen on scenarios.

Common WPA graphs for Modern Standby power management

The graphs that are generated from the Modern Standby WPA profile are key to observing system behavior in Modern Standby and identifying problems. Two commonly used WPA graphs are the Platform Idle State graph, which shows how much time the platform spends in the various platform idle states, and the DRIPS graph, which shows the activity levels of software and hardware components.

Each graph has a table view that shows the raw data that was used to construct the graph. The view can be configured by using the buttons that are located at the upper right-hand corner of the graph window.

The default view is graph only. The following paragraphs explain how to change the default view to obtain information about Modern Standby behavior.

Platform Idle State graph

The **Platform Idle State** graph shows the residency in platform idle states plotted against time.

On different platforms, the numerical states might correspond to different System on a Chip (SoC) states. Contact the SoC vendor to get the specific mapping for their hardware. This section covers only the lowest power platform state because the time spent in this state is critical to Modern Standby battery life.

The most important of the platform idle states is the deepest state, DRIPS. The DRIPS state corresponds to the lowest power state for the SoC during Modern Standby. Each SoC defines its own DRIPS state and corresponding state index.

The percentage of time that is spent in the DRIPS state (DRIPS percent) is an important metric for Modern Standby because it is directly proportional to battery life. If the DRIPS percent is high (above 90 percent), battery life will be longer than if the DRIPS percent is lower (for example, below 80 percent).

To obtain the DRIPS percent, open the table view and drag the % Duration column to filter on State. This column will then display the percentage of time that the system was in each state.

DRIPS graph

The DRIPS graph shows the components that are active during the trace period, including activators, devices, and processes. Use this graph to identify the components that are active the longest and that prevent the system from entering DRIPS.

Activators are components that take references and perform tasks while in Modern Standby. They handle the value-adding, explicitly allowed software activities that can run during sleep. Ideally, they should be active only in short bursts, and the DRIPS graph can be used to identify the most active activator during a Modern Standby session. This information is important because a particular activator could hold a reference for a long period of time, which prevents the system from entering DRIPS.

All components that are shown in the preceding graph, except for Devices and CPU Activity, are activators. For example, the preceding graph shows BI, WNS, NCSI, and Image Download Manager as activators. To identify the top activators, open the table view and look at the % Reason Time column, which shows the percentage of time the

activator was active during the Modern Standby session. For example, the following screenshot shows that BI is the top activator with 49.71 percent active.

BI is a special activator because it provides broker services to apps to access system resources. When BI shows up as an active activator, expand the BI row and determine which apps are causing BI to be active. Use this graph to determine the top active apps during the Modern Standby session.

In addition to activators, active devices might prevent the system from entering DRIPS.

Similar to system idle states, devices have low-power states that range from D0 to D3. Device low-power states are generally standardized by device class. Low-power states for devices the SoC itself are defined by the SoC manufacturer. Low-power states for devices outside of the SoC are typically standardized across all systems.

Use the DRIPS graph to determine the top active devices during the Modern Standby session. The graph shows only those devices that can block the SoC idle state (DRIPS), based on information provided by the platform power engine plug-in (PEP). For more information about the PEP, see PoFxPowerControl.

> **Note**
>
> Some devices can be active because an activator is running tasks that require the device to be active. Common examples are the primary storage (eMMc/SSD) and Wi-Fi devices, which are active whenever the BI activator is active.

To identify the most active devices, open the table view and look at the % Reason Time column, which shows the percentage of time that each device was active during the Modern Standby session.

In addition to activators and devices, a final reason that the system cannot enter DRIPS is due to excessive CPU activity. CPU activity is a less common problem compared to activators and devices, but might be exacerbated by OEM pre-installed desktop applications and services.

View the active processes by expanding the CPU Activity row.

Modern Standby FAQs

Article • 09/25/2020

Typical questions related to Modern Standby are answered here.

- **Q:** Which versions of Windows support Modern Standby?

  A: Modern Standby is supported on Windows 10 for desktop editions (Home, Pro, Enterprise, and Education) and Windows 10 Mobile. In Windows 8.0 and 8.1, the only supported standby power models were Connected Standby (CS) and S3.

- **Q:** Can I switch between S3 and Modern Standby by changing a setting in the BIOS?

  A: No, switching the power model is not supported in Windows without a complete OS re-install.

- **Q:** What if I have a platform that needs to support both Windows 10 and older versions of Windows?

  A: The following table describes what happens with platforms in each of a number of configurations.

  Note The S3 object is not required for Modern Standby support. However, if it is not available, S3 support won't be available.

Flags	Windows 7	Windows 8.1	Windows 10
ACPI_S0_LOW_POWER_IDLE FADT flag All built-in networking devices support WoL patterns, protocol offloads, and D0 packet coalescing ACPI firmware provides an S3 object	S3	Connected Standby	Modern Standby (Connected Standby-capable)
ACPI_S0_LOW_POWER_IDLE FADT flag ACPI firmware provides an S3 object	S3	S3	Disconnected Standby
ACPI firmware provides an S3 object	S3	S3	S3

Windows radial controller implementation guide

Article • 01/18/2019

This section is a guide for implementing a radial controller compatible with Windows 10.

In this section

Topic	Description
Windows Radial Controller Designs	This topic presents information about the radial controller devices that are compatible with Windows 10.
Device Bus Connectivity	This topic discusses bus connectivity methods for a Windows radial controller device.
Radial Controller Protocol Implementation	This topic provides information about radial controller power management, and discusses power consumption requirements.

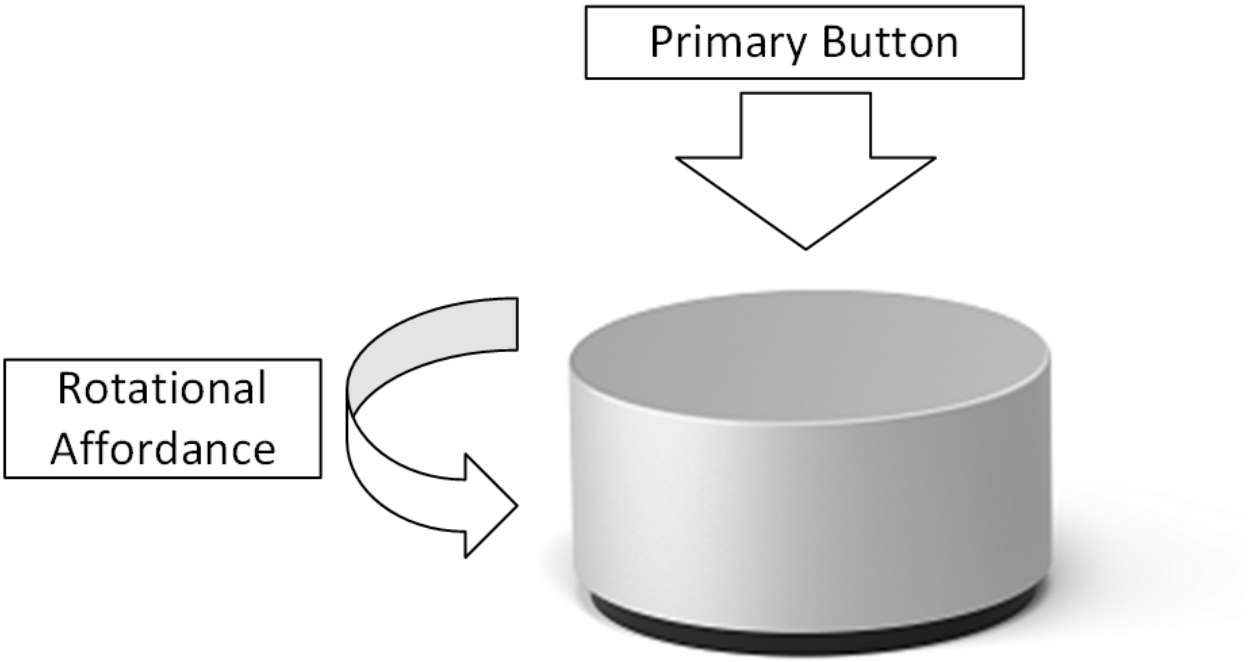
Windows radial controller designs

Article • 01/18/2019

This topic presents information about the radial controller devices that are compatible with Windows 10, version 1607.

Physical design

The physical radial controller device that the user interacts with may have a variety of industrial designs. However, all compatible Windows radial controller devices have a primary button, and a rotational affordance. The following image illustrates how these affordances may be implemented.



A radial controller may optionally report its on-screen position. The mechanism for determining the on-screen position is vendor-specific, and therefore a specific implementation is not outlined in this document. However, the method for reporting the on-screen position is standardized and detailed in the following sections.

Device bus connectivity

Article • 01/18/2019

This topic discusses bus connectivity methods for a Windows radial controller device.

A Windows radial controller may use the Microsoft provided inbox drivers to connect to its Windows host using the following:

- HID over Bluetooth
- HID over USB
- HID over I^2C

To utilize any other bus other than those listed above, a third-party HID mini-port driver is required. Microsoft strongly recommends using the Microsoft-provided inbox drivers.

We recommend using the Microsoft-provided inbox drivers. For devices that aggregate information from multiple input sources, a third-party HID filter driver may be required. If you elect to use a third-party mini-port or filter driver, the driver should be included in any OEM and system Restore images, and should also be made available through Windows Update.

Radial controller protocol implementation

Article • 11/05/2020

Windows radial controller devices are expected to use the Human Interface Device (HID) protocol to communicate with the host.

A good understanding of the HID protocol is needed to be able to understand the information presented here. See the following resources for information about the HID protocol:

- Device Class Definition for Human Interface Devices (HID)
- HID Usage Tables
- HID Over I2C Protocol Specification

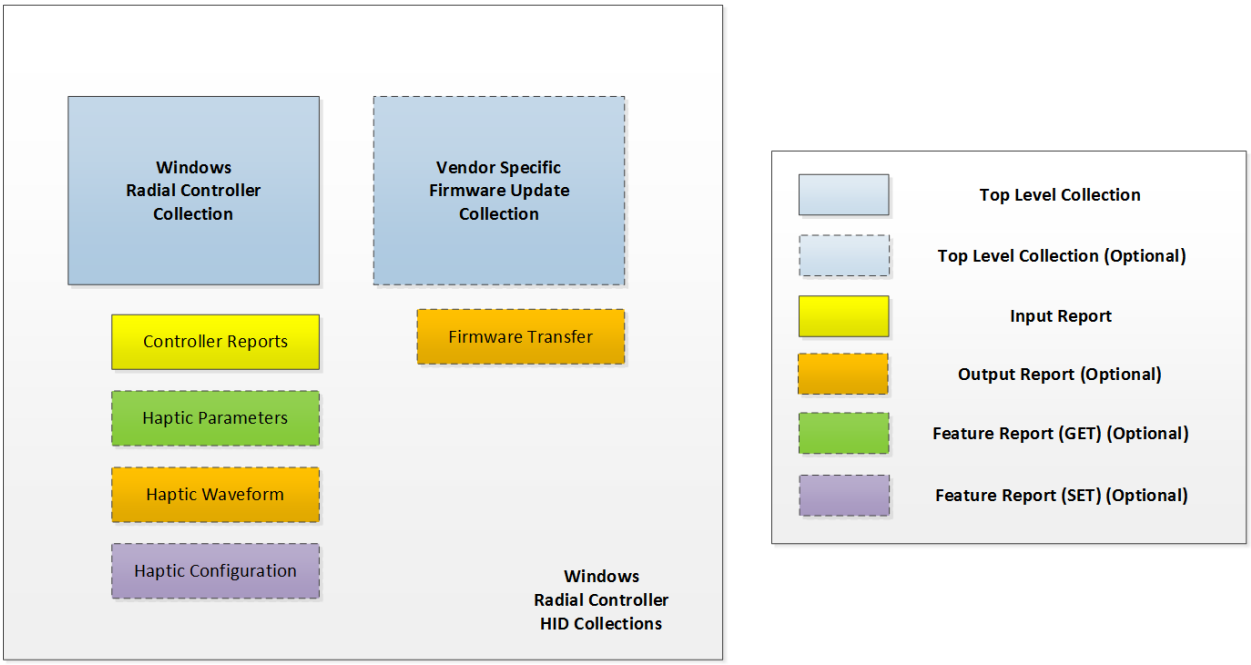
Windows includes a HID class driver and corresponding HID I^2C, HID USB and HID Bluetooth-enabled miniport drivers; therefore, there is no need for any third-party mini-port drivers for Windows radial controllers unless it utilizes a bus where an inbox HID mini-port driver is not available.

A device only needs to report the usages described in this topic in the firmware for a Windows radial controller device. Windows will use the firmware and its own HID drivers to enable the device and give Windows applications access to the device.

A sample descriptor is provided in the Sample Report Descriptors section.

Required HID top-level collections

A Windows radial controller device exposes, at a minimum, the single mandatory top-level collection, for radial controller input reporting. An optional (recommended) collection for firmware updates may also be implemented. The following diagram shows the HID collections for a Windows radial controller device.



Windows radial controller collection

Using the HID protocol, a Windows radial controller provides a top-level collection that appears as a System Multi-Axis Controller (Page **0x01**, Usage **0x0E**). By reporting this top-level collection and the required usages, a device appears to Windows as a radial controller.

The Windows radial controller collection provides input reporting to the host, as well as device information that pertains to those reports. The collection supports one mandatory input report for radial controller input data. An optional feature report and output report for haptic feedback may also be supported.

In this section

Topic	Description
Radial Controller Input Reports	This topic details the HID input reports that are used for radial controller haptic feedback support in Windows 10, version 1607 and later operating systems.
Radial Controller Output Reports	This topic details the HID output reports that are used for radial controller haptic feedback support in Windows 10, version 1607 and later operating systems.
Radial Controller Feature Reports	This topic details the HID feature reports that are used for radial controller haptic feedback support in Windows 10, version 1607 and later operating systems.

Topic	Description
Firmware Update Collection	This topic discusses an optional HID top-level collection that can be used for radial controller firmware updates in Windows 10, version 1607 and later operating systems.
Sample Report Descriptors	This topic presents a sample report descriptor and some sample descriptor excerpts for a Windows radial controller device.

Windows radial controller feature reports

Article • 03/17/2021

This topic details the HID feature reports that are used for radial controller haptic feedback support in Windows 10, version 1607 and later operating systems.

Haptic feedback collection (optional)

If a Windows radial controller device supports haptic feedback, it can allow the system and applications to take advantage of it by including a haptic feedback collection (Page **0x0E**, usage **0x01**) within the Windows radial controller TLC. For more information on how the HID specification supports haptic feedback, please consult the Haptics page ratification to the HID specification.

The host uses the following usages (through the haptic feedback collection) to communicate the haptic feedback waveform support and allow the host to configure the haptic feedback on the Windows radial controller device. If a device chooses to expose a haptic feedback collection, some usages are mandatory to allow automatic haptic feedback configuration to be supported.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Waveform List	Ordered list of haptic waveforms supported by the device	0x0E	0x10	Mandatory
Duration List	Ordered list of durations for waveforms in the Waveform List	0x0E	0x11	Mandatory
Auto Trigger	Waveform to fire automatically at device's discretion	0x0E	0x20	Mandatory
Auto Trigger Associated Control	HID usage of the control associated with haptic feedback	0x0E	0x22	Mandatory
Intensity	Output – Intensity of Manual Trigger waveform as a percentage	0x0E	0x23	Optional
Repeat Count	Output – Number of times to play Manual Trigger waveform after initial	0x0E	0x24	Optional

Member	Description	Page	ID	Mandatory/Optional
	play			
Retrigger Period	Output – Duration of time to wait before retriggering Manual Trigger when repeating	0x0E	0x25	Optional
Waveform Cutoff Time	Max time a Manual Trigger waveform can play before being cut off	0x0E	0x28	Optional

Waveform List

Waveform List usage represents a collection of the HID usages of supported waveforms, ordered using ordinals. The pre-defined haptics waveforms are defined in the HID specification. Windows supports implementing the following waveforms for a haptic feedback enabled device:

Expand table

Waveform	Description	Page	ID	Mandatory/Optional
WAVEFORM_NONE	No-op. Should not impact the play state of ongoing waveforms.	0x0E	0x1001	Mandatory
WAVEFORM_STOP	Stops the playback of ongoing waveforms.	0x0E	0x1002	Mandatory
WAVEFORM_CLICK	Creates a short “click” feedback	0x0E	0x1003	Mandatory
WAVEFORM_BUZZ_CONTINUOUS	Buzzes the device for an extended duration of time	0x0E	0x1004	Optional

Both WAVEFORM_NONE and WAVEFORM_STOP are required for all HID-compliant haptics devices. Ordinals 1 and 2 are implicitly set to WAVEFORM_NONE and WAVEFORM_STOP. They do not need to be declared in the Waveform List or Duration List. The Waveform List declares supported waveforms by the physical minimum and maximum of each ordinal in the list.

Duration List

Duration List usage represents a collection of the durations for the supported waveforms in the Waveform list, ordered using ordinals. The unit for waveform duration is milliseconds, and duration must be a positive non-zero value for any non-continuous waveform. If a waveform is continuous (will play until stopped by the host or the Waveform Cutoff Time is exceeded), then its duration is defined as zero.

WAVEFORM_NONE and WAVEFORM_STOP are assumed to have a duration of zero. They do not need to be declared in the Duration List.

Auto Trigger

The Auto Trigger usage represents a supported waveform usage from the Waveform List which will be triggered automatically by the device as needed based on user interactions. This allows lower-latency response to common user interactions like rotation and clicking. This may need to be modified or disabled by the host, and can be set via a feature report. The Auto Trigger waveform can be disabled by being set to zero by the host.

Auto Trigger Associated Control

The Auto Trigger Associated Control specifies the extended HID usage of the physical control associated with automatic haptic feedback on the device. For a radial controller device, this control should be defined as the Dial usage specified within the same top level collection.

Intensity

The Intensity usage represents the percentage of maximum intensity to apply to a waveform. This value should vary between 0 and 100 percent. 100 percent indicates waveforms will be triggered by the device at their maximum strength, and 0 percent indicates the haptic transducer is not enabled.

If defined in a feature report, it sets the default intensity for all Auto Trigger and Manual Trigger waveforms for the device.

Repeat Count

The Repeat Count usage represents the number of times to repeat a waveform. A Repeat Count of zero indicates the Manual Trigger waveform should only be played a single time (no repeat). If the Waveform Cutoff Time has been exceeded, it is expected that any incomplete repeats will be ignored.

If defined in a feature report, it sets the default repeat count for all Auto Trigger and Manual Trigger waveforms for the device.

Retrigger Period

The Retrigger Period usage represents the amount of time for the device to wait before repeating a Manual Trigger waveform in an output report, per the value specified by Repeat Count. The units for this value are milliseconds. If the Retrigger Period is smaller than the duration of the waveform being played, the waveform should be stopped and restarted at the time period indicated by the retrigger period.

If defined in a feature report, it sets the default retrigger period for all Auto Trigger and Manual Trigger waveforms for the device.

Waveform Cutoff Time

The Waveform Cutoff Time usage represents the maximum amount of time the device will allow a Manual Trigger waveform to be repeated before ending playback. This is a constant value for the device and includes both continuous waveforms without set durations, and waveforms with discrete durations which are set to repeat many times. The units for this value are milliseconds.

Windows radial controller firmware update collection (optional)

Article • 03/17/2021

This topic discusses an optional HID top-level collection that can be used for radial controller firmware updates in Windows 10, version 1607 and later operating systems.

A Windows radial controller device can use the HID protocol to provide a vendor-specific top-level collection for performing device firmware, and vendor configuration updates.

The vendor-specific firmware update collection can provide an output report for transferring the firmware payload from the host to the device.

This is highly advantageous as it allows for firmware updates to be performed without requiring a driver on the host. It is mandatory for the *wVersionID* device attribute to be incremented after a firmware upgrade.

A Windows radial controller device should be able to recover from a firmware update that failed due to power loss (or some other error), if the device is powered down, and then its power supply is reapplied. It is highly recommended that basic functionality be available even after a failed firmware update.

Windows radial controller input reports

Article • 03/17/2021

This topic details the HID input report that are used for radial controller haptic feedback support in Windows 10, version 1607 and later operating systems.

The host makes use of the following usages when extracting data from an input report:

Member	Description	Page	ID	Mandatory/Optional
Button	State of the button located on radial controller	0x09	0x01	Mandatory
Dial	Relative rotation of the radial controller	0x01	0x37	Mandatory
X	X coordinate of contact position	0x01	0x30	Optional
Y	Y coordinate of contact position	0x01	0x31	Optional
Width	Width of bounding box around a contact	0x0D	0x48	Optional
Height	Height of bounding box around a contact	0x0D	0x49	Optional

Mandatory and Windows-supported optional usages

Any device that does not report all mandatory usages will be non-functional as a Windows radial controller. Mandatory usages are strictly enforced by the Windows host. Where a logical maximum value has not been restricted, it can be optimized to reduce descriptor size.

The following sections provide more details about the mandatory and the Windows-supported optional usages that were presented in the above table.

Button

Used to indicate when the primary button on the radial controller device is depressed. This is indicated by a main item with a report size of 1 bit. When delivering an input report, the bit should be set when this button is depressed, and cleared when the button is released.

Dial

Used to indicate the clockwise rotation of the radial controller around its own major axis. When delivering an input report, the reported value should be the magnitude of rotation relative to the previous report. This value will be positive if the radial controller is rotated in a clockwise direction, and negative if the radial controller is rotated in a counterclockwise direction.

The physical range and logical range must be specified. Degrees or radians can be used for the physical range. In either case, the logical range must be large enough to report values that are accurate to at least one decimal place.

X/Y

X and Y report the coordinates of a radial controller's contact on-screen. This coordinate represents the center of the contact relative to the display.

Devices that report X and Y must also report a Width or Height usage. The host uses the reported coordinate to build the bounding rectangle around the contact.

The following global items must be specified for the X and Y usages:

- Logical minimum
- Logical maximum
- Physical minimum
- Physical maximum
- Unit
- Unit exponent

The physical range for the device and the units must be accurately reported. If the information is inaccurate, the device will not work correctly. Devices must also report data within the logical range that is specified in the report descriptor.

X and Y can provide a NULL value to the system, implying the device does not currently have a position relative to a display, by providing a value outside the reported logical range and adding Null Support to the usage's input report **(Bit 6 {No Null position (0) | Null state(1)}).**

Width and height

The width and height usages represent the width and height of the bounding box around the radial controller contact. The width and height should be defined as constants in the report descriptor.

A Windows radial controller device is assumed to be circular by the system. Because of this, it is not required to report both width and height for an onscreen contact. If either height or width is supported, the bounds of the contact will be interpreted by the system correctly. Windows does not provide support for radial controller devices with different values for width and height.

Windows radial controller output reports

Article • 03/17/2021

This topic details the HID output reports that are used for radial controller haptic feedback support in Windows 10, version 1607 and later operating systems.

Haptic feedback collection (optional)

If a Windows radial controller device supports haptic feedback, it can allow the system and applications to take advantage of it by including a haptic feedback collection (Page **0x0E**, Usage **0x01**) within the Windows radial controller TLC. For more information on how the HID specification supports haptic feedback, please consult the Haptics page ratification to the HID specification.

The host uses the following usages in an output report (through the haptic feedback collection) to allow the host to issue haptic feedback events to the Windows radial controller device. If a device chooses to expose a haptic feedback collection, some usages are mandatory to allow host-initiated haptic feedback to be supported.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Manual Trigger	Waveform to fire as explicit command from the host.	0x0E	0x21	Mandatory
Intensity	Output – Intensity of Manual Trigger waveform as a percentage	0x0E	0x23	Optional
Repeat Count	Output – Number of times to play Manual Trigger waveform after initial play	0x0E	0x24	Optional
Retrigger Period	Output – Duration of time to wait before retriggering Manual Trigger when repeating	0x0E	0x25	Optional

Manual trigger

The Manual Trigger usage represents a supported waveform usage from the waveform list which has been requested to be played by the host. When an output report containing a Manual Trigger other than WAVEFORM_NONE is sent to the device, it

should immediately begin playing the specified waveform with the additional properties included in the output report (Intensity, Repeat Count, Retrigger Period). When an output report contains a Manual Trigger of WAVEFORM_STOP, any ongoing waveform playback should be stopped.

Intensity

The Intensity usage represents the percentage of maximum intensity to apply to a waveform. This value should vary between 0 and 100 percent. 100 percent indicates waveforms will be triggered by the device at their maximum strength, and 0 percent indicates the haptic transducer is not enabled.

When set in an output report, this value will override the default Intensity for the Manual Trigger waveform defined in the same output report. This does not set a new default Intensity.

Repeat count

The Repeat Count usage represents the number of times to repeat a waveform. A Repeat Count of zero indicates the Manual Trigger waveform should only be played a single time (no repeat). If the waveform cutoff time has been exceeded, it is expected that any incomplete repeats will be ignored.

When set in an output report, this value overrides the default repeat count for the Manual Trigger waveform defined in the same output report. This does not set a new default Repeat Count.

Retrigger Period

The Retrigger Period usage represents the amount of time for the device to wait before repeating a Manual Trigger waveform in an output report, per the value specified by Repeat Count. The units for this value are milliseconds. If the Retrigger Period is smaller than the duration of the waveform being played, the waveform should be stopped and restarted at the time period indicated by the retrigger period.

When set in an output report, it overrides the default retrigger period for the Manual Trigger waveform defined in the same output report. This does not set a new default Retrigger Period.

Windows radial controller sample report descriptors

Article • 12/05/2022

This topic presents a sample report descriptor and some sample descriptor excerpts for a Windows radial controller device.

> **Note**
>
> Although vendor specific usages may be used at any level of the report, they must be used in a way that could be implemented equally across all reporting methods, namely, serial, parallel, and hybrid. Not adhering to this guidance may not result in an HLK failure but could result in serious compatibility issues.

Sample input report descriptor

The following is a sample report descriptor for an integrated Windows radial controller top-level collection.

```
// Integrated Radial Controller TLC
0x05, 0x01,     // USAGE_PAGE (Generic Desktop)
0x09, 0x0e,     // USAGE (System Multi-Axis Controller)
0xa1, 0x01,     // COLLECTION (Application)
0x85, 0x01,     //   REPORT_ID (Radial Controller)
0x05, 0x0d,     //   USAGE_PAGE (Digitizers)
0x09, 0x21,     //   USAGE (Puck)
0xa1, 0x00,     //   COLLECTION (Physical)
0x05, 0x09,     //     USAGE_PAGE (Buttons)
0x09, 0x01,     //     USAGE (Button 1)
0x95, 0x01,     //     REPORT_COUNT (1)
0x75, 0x01,     //     REPORT_SIZE (1)
0x15, 0x00,     //     LOGICAL_MINIMUM (0)
0x25, 0x01,     //     LOGICAL_MAXIMUM (1)
0x81, 0x02,     //     INPUT (Data,Var,Abs)
0x05, 0x01,     //     USAGE_PAGE (Generic Desktop)
0x09, 0x37,     //     USAGE (Dial)
0x95, 0x01,     //     REPORT_COUNT (1)
0x75, 0x0f,     //     REPORT_SIZE (15)
0x55, 0x0f,     //     UNIT_EXPONENT (-1)
0x65, 0x14,       //     UNIT (Degrees, English Rotation)
0x36, 0xf0, 0xf1,   //     PHYSICAL_MINIMUM (-3600)
0x46, 0x10, 0x0e,   //     PHYSICAL_MAXIMUM (3600)
0x16, 0xf0, 0xf1,   //     LOGICAL_MINIMUM (-3600)
```

```
0x26, 0x10, 0x0e,   //     LOGICAL_MAXIMUM (3600)
0x81, 0x06,     //     INPUT (Data,Var,Rel)
0x09, 0x30,     //     USAGE (X)
0x75, 0x10,     //     REPORT_SIZE (16)
0x55, 0x0d,     //     UNIT_EXPONENT (-3)
0x65, 0x13,     //     UNIT (Inch,EngLinear)
0x35, 0x00,     //     PHYSICAL_MINIMUM (0)
0x46, 0xc0, 0x5d,   //     PHYSICAL_MAXIMUM (24000)
0x15, 0x00,     //     LOGICAL_MINIMUM (0)
0x26, 0xff, 0x7f,   //     LOGICAL_MAXIMUM (32767)
0x81, 0x02,     //     INPUT (Data,Var,Abs)
0x09, 0x31,     //     USAGE (Y)
0x46, 0xb0, 0x36,   //     PHYSICAL_MAXIMUM (14000)
0x81, 0x02,     //     INPUT (Data,Var,Abs)
0x05, 0x0d,     //     USAGE_PAGE (Digitizers)
0x09, 0x48,     //     USAGE (Width)
0x36, 0xb8, 0x0b,   //     PHYSICAL_MINIMUM (3000)
0x46, 0xb8, 0x0b,   //     PHYSICAL_MAXIMUM (3000)
0x16, 0xb8, 0x0b,   //     LOGICAL_MINIMUM (3000)
0x26, 0xb8, 0x0b,   //     LOGICAL_MAXIMUM (3000)
0x81, 0x03      //     INPUT (Cnst,Var,Abs)
0xc0,           //   END_COLLECTION
0xc0,           // END_COLLECTION
```

Sample haptic feedback feature report descriptor

The following is a report descriptor snippet for a haptic feedback collection to be included in an integrated Windows radial controller top-level collection.

```
0x85, 0x02,           // REPORT_ID (Haptic Feedback)
0x05, 0x0e,           // USAGE_PAGE (Haptics)
0x09, 0x01,           // USAGE (Simple Haptic Controller)
0xa1, 0x02,           // COLLECTION (Logical)
0x09, 0x20,           //   USAGE (Auto Trigger)
0x16, 0x00, 0x10,       //   LOGICAL_MINIMUM (0x1000)
0x26, 0x04, 0x10,       //   LOGICAL_MAXIMUM (0x1004)
0x95, 0x01,           //   REPORT_COUNT (1)
0x75, 0x10,           //   REPORT_SIZE (16)
0xb1, 0x02,           //   FEATURE (Data,Var,Abs)
0x09, 0x21,           //   USAGE (Manual Trigger)
0x91, 0x02,           //   OUTPUT (Data,Var,Abs)
0x09, 0x22,           //   USAGE (Auto Trigger Associated Control)
0x17, 0x37, 0x00, 0x01, 0x00,     //   LOGICAL_MINIMUM (0x00010037)
0x27, 0x37, 0x00, 0x01, 0x00,    //   LOGICAL_MAXIMUM (0x00010037)
0x95, 0x01,           //   REPORT_COUNT (1)
0x75, 0x20,           //   REPORT_SIZE (32)
0xb1, 0x03,           //   FEATURE (Cnst,Var,Abs)
```

```
0x09, 0x23,            //   USAGE (Intensity)
0x15, 0x00,            //   LOGICAL_MINIMUM (0)
0x25, 0x7f,            //   LOGICAL_MAXIMUM (127)
0x75, 0x08,            //   REPORT_SIZE (8)
0x91, 0x02,            //   OUTPUT (Data,Var,Abs)
0x09, 0x23,            //   USAGE (Intensity)
0xb1, 0x02,            //   FEATURE (Data,Var,Abs)
0x09, 0x24,            //   USAGE (Repeat Count)
0x91, 0x02,            //   OUTPUT (Data,Var,Abs)
0x09, 0x24,            //   USAGE (Repeat Count)
0xb1, 0x02,            //   FEATURE (Data,Var,Abs)
0x09, 0x25,            //   USAGE (Retrigger Period)
0x91, 0x02,            //   OUTPUT (Data,Var,Abs)
0x09, 0x25,            //   USAGE (Retrigger Period)
0xb1, 0x02,            //   FEATURE (Data,Var,Abs)
0x09, 0x28,            //   USAGE (Waveform Cutoff Time)
0x26, 0xff, 0x7f,        //   LOGICAL_MAXIMUM (32,767)
0x75, 0x10,            //   REPORT_SIZE (16)
0xb1, 0x02,            //   FEATURE (Data,Var,Abs)
0x05, 0x0e,            //   USAGE_PAGE (Haptics)
0x09, 0x10,            //   USAGE (Waveform List)
0xa1, 0x02,            //   COLLECTION (Logical)
0x05, 0x0A,            //     USAGE_PAGE (Ordinal)
0x09, 0x03,            //     USAGE (Ordinal 3)
0x95, 0x01,            //     REPORT_COUNT (1)
0x75, 0x08,            //     REPORT_SIZE (8)
0x15, 0x03,            //     LOGICAL_MINIMUM (3)
0x25, 0x03,            //     LOGICAL_MAXIMUM (3)
0x36, 0x03, 0x10,        //     PHYSICAL_MINIMUM (0x1003)
0x46, 0x03, 0x10,        //     PHYSICAL_MAXIMUM (0x1003)
0xb1, 0x03,            //     FEATURE (Cnst,Var,Abs)
0x09, 0x04,             //     USAGE (Ordinal 4)
0x15, 0x04,             //     LOGICAL_MINIMUM (4)
0x25, 0x04,            //     LOGICAL_MAXIMUM (4)
0x36, 0x04, 0x10,        //     PHYSICAL_MINIMUM (0x1004)
0x46, 0x04, 0x10,        //     PHYSICAL_MAXIMUM (0x1004)
0xb1, 0x03            //     FEATURE (Cnst,Var,Abs)
0xc0,                //   END_COLLECTION
0x05, 0x0e,            //   USAGE_PAGE (Haptics)
0x09, 0x11,            //   USAGE (Duration List)
0xa1, 0x02,            //   COLLECTION (Logical)
0x05, 0x0a            //     USAGE_PAGE (Ordinal)
0x09, 0x03,            //     USAGE (Ordinal 3)
0x09, 0x04,            //     USAGE (Ordinal 4)
0x15, 0x00,            //     LOGICAL_MINIMUM (0)
0x26, 0xff, 0x0f,        //     LOGICAL_MAXIMUM (4095)
0x95, 0x02,            //     REPORT_COUNT (2)
0x75, 0x08,            //     REPORT_SIZE (8)
0xb1, 0x02,            //     FEATURE (Data,Var,Abs)
0xc0,                //   END_COLLECTION
```

Security considerations for Original Equipment Manufacturers

Article • 02/08/2023

As an Original Equipment Manufacturer (OEM), you have a unique opportunity to impact the efficacy of the security measures available to your customers. Customers want and need the ability to secure their devices. Windows 10 security features are built on top of security-enabled hardware and firmware. That's where you come in. To provide a differentiator for your devices, or to sell in the Enterprise space, you want to provide the latest hardware enhancements, with which Windows 10 can be configured for safety.

IT Professionals: To learn more about these features including how to deploy them in your enterprise, see Device Security and Control the health of Windows 10-based devices.

Windows 10 S

Windows 10 S is a specific configuration of Windows 10 Pro that offers a familiar Windows experience that's streamlined for security and performance. Windows 10 S provides the best of the cloud and full featured apps, and is designed for modern devices. Microsoft Defender is always on and always up-to-date.

Windows 10 S will only run verified apps from the Store and verified drivers from Windows Update. Windows 10 S provides supports Azure Active Directory, and when paired with MSA or Intune for Education, Windows 10 S defaults to storing files to OneDrive.

OEMs: For more information about Windows 10 S, see Windows 10 S security features and requirements for OEMs.

BitLocker device encryption

BitLocker device encryption is a set of features enabled by an OEM by providing the right set of hardware in the devices you sell. Without the proper hardware configuration, device encryption is not enabled. With the right hardware configurations, Windows 10 automatically encrypts a device.

OEMs: To learn more about BitLocker, see BitLocker drive encryption in Windows 10 for OEMs.

Secure Boot

Secure Boot is a security standard developed by members of the PC industry to help make sure that a PC boots using only software that is trusted by the PC manufacturer. When the PC starts, the firmware checks the signature of each piece of boot software, including firmware drivers (Option ROMs), EFI applications, and the operating system. If the signatures are valid, the PC boots, and the firmware gives control to the operating system.

OEMs: To learn more about Secure Boot requirements for OEMs, see Secure Boot.

Trusted Platform Module (TPM) 2.0

Trusted Platform Module (TPM) technology is designed to provide hardware-based, security-related functions. A TPM chip is a secure crypto-processor that helps you with actions such as generating, storing, and limiting the use of cryptographic keys. The chip includes multiple physical security mechanisms to make it tamper resistant, and malicious software is unable to tamper with the security functions of the TPM.

> **Note**
>
> Since July 28, 2016, all new device models, lines or series (or if you are updating the hardware configuration of an existing model, line or series with a major update, such as CPU, graphic cards) must implement and enable by default TPM 2.0 (details in section 3.7 of the Minimum hardware requirements page). The requirement to enable TPM 2.0 only applies to the manufacturing of new devices.

OEMs: For more information, see Trusted Platform Module (TPM) 2.0 hardware requirements.

IT Professionals: To understand how TPM works in your enterprise, see Trusted Platform Module

Unified Extensible Firmware Interface (UEFI) requirements

UEFI is a replacement for the older BIOS firmware interface. When the devices starts, the firmware interface controls the booting process of the PC, and then passes control to Windows or another operating system. UEFI enables security features such as Secure Boot and factory encrypted drives that help prevent untrusted code from running before

the operating system is loaded. As of Windows 10, version 1703, Microsoft requires UEFI Specification version 2.3.1c. To learn more about the OEM requirements for UEFI, see UEFI firmware requirements.

OEMs: To learn more about what you need to do in order to support UEFI drivers, see UEFI in Windows.

Virtualization-based Security (VBS)

Hardware-based security features, also called virtualization-based security or VBS, provides isolation of secure kernel from normal operating system. Vulnerabilities and Zero-Day attacks in the operating system cannot be exploited because of this isolation.

OEMs: For more information about VBS hardware requirements, see Virtualization Based Security (VBS) hardware requirements.

Microsoft Defender Application Guard

Application Guard helps to isolate enterprise-defined untrusted sites, protecting an enterprise while its employees browse the Internet.

If you are selling devices to enterprise customers, you want to provide hardware that supports the security features that enterprises need.

OEMs: To learn more about hardware requirements for Microsoft Defender Application Guard, see Microsoft Defender Application Guard hardware requirements.

Microsoft Defender Credential Guard

Credential Guard uses virtualization-based security to isolate and protect secrets (for example, NTLM password hashes and Kerberos ticket-granting tickets) to block pass-the-hash or pass-the-ticket attacks.

OEMs: To learn more about hardware requirements for Microsoft Defender Credential Guard, see Microsoft Defender Credential Guard hardware requirements.

IT Professionals: To learn how to configure and deploy Microsoft Defender Credential Guard in your enterprise, see Protect derived domain credentials with Microsoft Defender Credential Guard.

Hypervisor-Protected Code Integrity is a combination of enterprise-related hardware and software security features that, when configured together, will lock a device down

so that it can only run trusted applications that are defined in code integrity policies.

Starting in Windows 10, 1703, the Microsoft Defender Device Guard features have been grouped into two new features: **Microsoft Defender Exploit Guard** and **Microsoft Defender Application control**. When these are both enabled, Hypervisor-Protected Code Integrity is enabled.

OEMs: For more information about Hypervisor-Protected Code Integrity hardware requirements, see Virtualization-based Security (VBS).

IT Professionals: To learn how to deploy Hypervisor-Protected Code Integrity in your enterprise, see Requirements and deployment planning guidelines for Hypervisor-Protected Code Integrity.

Kernel DMA Protection

Hardware-based security features, also called Memory Access Protection, provides isolation and protection against malicious DMA attacks during the boot process and during the runtime of the OS.

OEMs: For more information about Kernel DMA Protection platform requirements, see Kernel DMA Protection for OEMs.

Driver Developers: For more information on Kernel DMA Protection and DMA Remapping compatible drivers, see Enabling DMA Remapping for device drivers.

IT Professionals: To learn more about Kernel DMA Protection policies and user experience, see Kernel DMA Protection.

Windows Hello

Microsoft Windows Hello gives users a personal, secured experience where the device is authenticated based on their presence. Users can log in with a look or a touch, with no need for a password. In conjunction with Microsoft Passport, biometric authentication uses fingerprints or facial recognition and is more secure, more personal, and more convenient.

For information about how Windows Hello works with the Companion Device Framework, see Windows Hello and the Companion Device Framework.

For information on biometric requirements for supporting Windows Hello, see Windows Hello biometric requirements.

For information about how face authentication works, see Windows Hello face authentication.

Windows 10 S security features and requirements for OEMs

Article • 04/03/2023

Windows 10 S is a specific configuration of Windows 10 Pro that offers a familiar Windows experience that's streamlined for security and performance. Windows 10 S provides the best of the cloud and full featured apps, and is designed for modern devices. Microsoft Defender is always on and always up-to-date.

Windows 10 S will only run verified apps from the Store and verified drivers from Windows Update. Windows 10 S provides supports Azure Active Directory, and when paired with MSA or Intune for Education, Windows 10 S defaults to storing files to OneDrive.

Features enabled for Windows 10 S

Windows 10 S Mode protects customers by using a combination of code integrity policies, hardware, and certification for apps. Windows 10 S will only run executable code that is signed with a Windows, WHQL, ELAM, or Store certificate from the Windows Hardware Developer Center Dashboard. This includes companion apps for drivers.

Features	Windows 10 S	Windows 10 Home	Windows 10 Pro
Non-store apps		Yes	Yes
Domain join on premise			Yes
Azure AD domain join	Yes		Yes
Windows Store apps (including Win32 centennial apps)	Yes	Yes	Yes
OneDrive automatic setup and sync; Requires MSA	Yes	Configurable	Configurable
Microsoft default apps set	Yes	Configurable	Configurable
Windows update for business	Yes		Yes
Windows Store for business	Yes		Yes
Mobile Device Management (MDM)	Yes	limited	Yes

Features	Windows 10 S	Windows 10 Home	Windows 10 Pro
BitLocker	Yes		Yes
Enterprise state roaming with Azure AD	Yes		Yes
Shared PC configuration	Yes		Yes

Windows 10 S default modern app configuration

- Email: Mail
- Maps : Maps
- Photo viewer : Photos
- Search : Bing
- Video player: Movies & TV
- Web browser: Edge
- OneDrive automatically configured for MSA accounts so that documents, Photos, and Desktop are automatically synced and the user has 5GB of standard storage.

Memory integrity protection

Memory integrity is a virtualization-based security (VBS) feature available in Windows 10, Windows 11, and Windows Server 2016 or higher. Memory integrity and VBS improve the threat model of Windows and provide stronger protections against malware trying to exploit the Windows kernel. VBS uses the Windows hypervisor to create an isolated virtual environment that becomes the root of trust of the OS that assumes the kernel can be compromised. Memory integrity is a critical component that protects and hardens Windows by running kernel mode code integrity within the isolated virtual environment of VBS. Memory integrity also restricts kernel memory allocations that could be used to compromise the system, ensuring that kernel memory pages are only made executable after passing code integrity checks inside the secure runtime environment, and executable pages themselves are never writable.

> **Note**
>
> Memory integrity is sometimes referred to as *hypervisor-protected code integrity (HVCI)* or *hypervisor enforced code integrity*, and was originally released as part of *Device Guard*. Device Guard is no longer used except to locate memory integrity and VBS settings in Group Policy or the Windows registry.

Memory integrity is turned on by default on clean installs of Windows 10 in S mode and Windows 11 on compatible hardware as described in memory integrity enablement. On other systems that don't meet the memory integrity auto-enablement requirements, customers can opt in using any of the methods described in how to enable memory integrity.

Kernel mode code integrity, protected by memory integrity, prevents the execution of unsigned or improperly signed binaries in the kernel. Using unsupported binaries should only be done during lab or factory image customization, or during deployment where the execution environment is either WinPE or Audit Mode.

For more information, see Memory integrity and virtualization-based security

User mode code integrity policy

Windows 10 in S mode is implemented using a policy that enforces user mode code integrity (CI). Once the CI policy is enabled on a system, it is enabled in two places:

- Windows 10 S, enforced at boot
- UEFI firmware policy, enforced during firmware load and OS boot

Signed drivers and Windows 10 S

Driver signing is different for Windows 10 S. To install on Windows 10 S, driver packages must meet the following requirements:

- Driver packages must be digitally signed with a Windows, WHQL, ELAM, or Store certificate from the Windows Hardware Developer Center Dashboard.
- Companion software must be signed with a Microsoft Store Certificate.
- Does not include an *.exe, *.zip, *.msi or *.cab in the driver package that extracts unsigned binaries.
- Driver installs using only INF directives.
- Driver does not call blocked inbox components.
- Drivers does not include any user interface components, apps, or settings. Instead, use Universal applications from the Microsoft Store, for example:
  - Hardware Support Apps
  - UWP device apps
  - Centennial Apps
  - Driver and firmware servicing uses Windows Update and not an updater app.

For more information, see Windows 10 S Driver Requirements and Publish a driver to Windows Update.

What's not supported

Windows 10 S does not allow any apps that aren't in the Store. A second limitation is that Windows 10 S does not allow on-premise domain joins. Additionally, some Windows customizations and some apps are not supported. For more information, see Planning a Windows 10 S deployment

The following components are blocked from running in Windows 10 S. Any script or application that calls one of these blocked components will be blocked. If your manufacturing process uses scripts or applications that rely on blocked components, you can temporarily enable manufacturing mode for configuring and testing, but you can't ship a PC with manufacturing mode enabled.

- bash.exe
- cdb.exe
- cmd.exe
- cscript.exe
- csi.exe
- dnx.exe
- kd.exe
- lxsmanager.dll
- msbuild.exe
- ntsd.exe
- powershell.exe
- powershell_ise.exe
- rcsi.exe
- reg.exe
- regedt32.exe
- windgb.exe
- wmic.exe
- wscript.exe

Windows 11 Secured-core PCs

Article • 05/06/2024

Microsoft works closely with OEM partners to help ensure that all certified Windows systems deliver a secure operating environment. Windows integrates closely with the hardware to deliver protections that take advantage of available hardware capabilities:

- Baseline Windows security – recommended baseline for all individual systems that provides foundational system integrity protections. Leverages TPM 2.0 for a hardware root of trust, secure boot and BitLocker drive encryption.
- Virtualization-based security enabled – leverages virtualization capabilities from hardware and the hypervisor to provide additional protection for critical subsystems and data.
- Secured-core – recommended for the most sensitive systems and industries like financial, healthcare, and government agencies. Builds on the previous layers and leverages advanced processor capabilities to provide protection from firmware attacks.

Secured-core PCs

Microsoft is working closely with OEM partners and silicon vendors to build Secured-core PCs that features deeply integrated hardware, firmware and software to ensure enhanced security for devices, identities and data.

Secured-core PCs provide protections that are useful against sophisticated attacks and can provide increased assurance when handling mission-critical data in some of the most data-sensitive industries, such as healthcare workers that handle medical records and other personally identifiable information (PII), commercial roles that handle high business impact and highly sensitive data, such as a financial controller with earnings data.

For general purpose laptops, tablets, 2-in-1's, mobile workstations, and desktops, Microsoft recommends using Security baselines for optimal configuration. For more info, see Windows security baselines.

Baseline Windows security is supported by Secure Boot, Bitlocker device encryption, Microsoft Defender, Windows Hello and a TPM 2.0 chip to provide a hardware root of trust for the OS platform. These features are designed to secure general purpose modern devices. If you are a decision maker purchasing new devices, your devices should meet the baseline Windows security requirements.

What makes a Secured-core PC

Expand table

Benefit	Feature	Hardware/Firmware requirement	Baseline Windows Security	Secured-core PCs
Create a hardware backed root of trust				
Secure Boot	Secure Boot is enabled in the BIOS by default.	✅	✅	✅
Secure Boot	3rd party UEFI CA not trusted by default, with BIOS option for enabling trust			✅
Trusted Platform Module 2.0 (TPM)	Meet the latest Microsoft requirements for the Trusted Computing Group (TCG) specification	✅	✅	✅
Direct Memory Access (DMA) Protection	The device supports Memory Access Protection (Kernel DMA Protection)	✅	✅	✅
Defend against firmware level attacks (either of the 2 approaches specified can be used)	System Guard Secure Launch (D-RTM) with System Management Mode (SMM) isolation	Enabled on device (via Secure Launch)	✅	✅
S-RTM and Standalone MM with MM supervisor (the approach	Supported on devices that have the FASR firmware		✅	

Benefit	Feature	Hardware/Firmware requirement	Baseline Windows Security	Secured-core PCs
implemented on FASR devices)				
Protect the OS from execution of unverified code	Hypervisor Code Integrity (HVCI)	Enabled on device	✅	✅
Provide advanced identity verification and protection	Windows Hello with Enhanced Sign-in Security (ESS)	A device with *Windows Hello with ESS* is enabled if it has the ESS hardware built-in components for face or fingerprint authentication, and the necessary support in BIOS	✅*	✅
Protect critical data if a device is lost, stolen or confiscated	BitLocker encryption	BitLocker can leverage the TPM 2.0 to encrypt and protect data	✅	

*Only possible on devices that have built-in Windows Hello biometric sign-in face or fingerprint sensors.

How to restore Windows 11 Secured-core PC configurations for Enterprise Administrators

Article • 07/25/2022

This document provides the steps to restore Secured-core PC configuration settings in the scenario where an Enterprise customer reimages a Secured-core PC, and subsequently needs to reenable all the Secured-core PC features. In sum, this document will cover the steps to enable the following Secured-core PC features, which can also be found on the Windows 11 Secured-core PCs webpage:

- System Management Mode (SMM) Protection
- Memory Access Protection enabled
- Enhanced Sign-in Security capable
- Memory Integrity (HVCI) enabled
- Trusted Platform Module 2.0
- Modern Device Servicing
- UEFI Secure Boot enabled
- Firmware Virtualization enabled

Managing Secured-core PC settings

Broadly, the settings for a Secured-core PC can be broken into two categories: those that can be configured via Windows, and those that are configured via the device UEFI firmware. However, even in these categories, some of the Windows settings require the device firmware to have specific settings. This document will explain how to manage these settings from these two perspectives and will call-out any firmware settings required to support those settings.

Windows settings

Windows Defender System Guard and SMM Protection

To protect critical resources such as the Windows authentication stack, single sign-on tokens, the Windows Hello biometric stack, and the Virtual Trusted Platform Module, a system's firmware and hardware must be trustworthy. Windows Defender System Guard (WDSG) reorganizes the existing Windows 11 system integrity features under one roof

and sets up the next set of investments in Windows security. It's designed to make these security guarantees:

- Protect and maintain the integrity of the system as it starts up
- Validate that system integrity has truly been maintained through local and remote attestation

Additionally, WDSG also protects against System Management Mode (SMM) attacks. SMM is a special-purpose CPU mode in x86 microcontrollers that handles power management, hardware configuration, thermal monitoring, and anything else the manufacturer deems useful. Whenever one of these system operations is requested, a non-maskable interrupt (SMI) is invoked at runtime, which executes SMM code installed by the BIOS. SMM code executes in the highest privilege level and is invisible to the OS, which makes it an attractive target for malicious activity. Even if System Guard Secure Launch is used to late launch, SMM code can potentially access hypervisor memory and change the hypervisor.

To defend against this, two techniques are used:

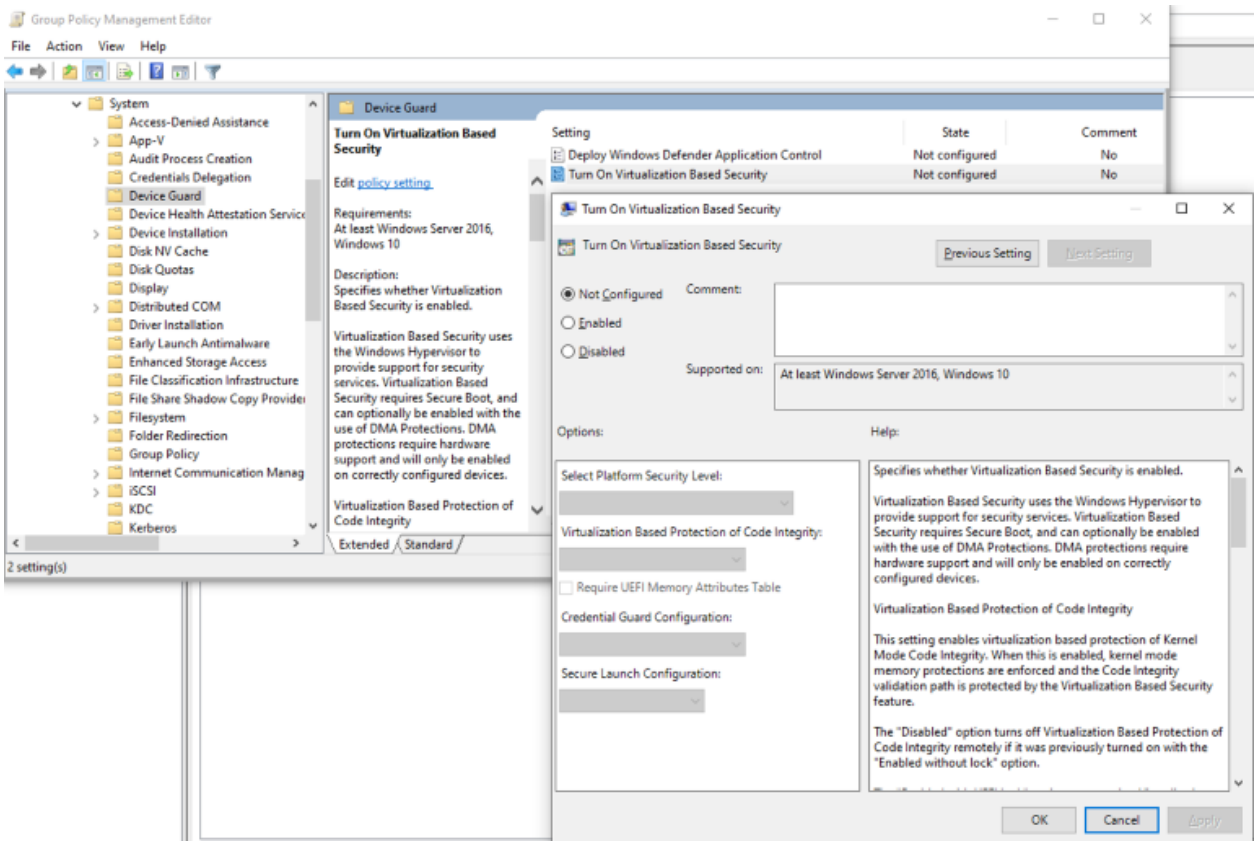
- Paging protection to prevent inappropriate access to code and data
- SMM hardware supervision and attestation

Paging protection can be implemented to lock certain code tables to be read-only to prevent tampering. This prevents access to any memory that has not been specifically assigned.

A hardware-enforced processor feature known as a supervisor SMI handler can monitor SMM and make sure it does not access any part of the address space that it is not supposed to.

SMM protection is built on top of the Secure Launch technology and requires it to function. In the future, Windows 11 will also measure this SMI Handler's behavior and attest that no OS-owned memory has been tampered with.

Reenabling WDSG protection can be done this in several ways. For example, to enable WDSG protection using GPO you would deploy the “Turn on Virtualization Based Security” template and set the Secure Launch Configuration:


Group Policy Management Editor
File Action View Help
System
Access-Denied Assistance
App-V
Audit Process Creation
Credentials Delegation
Device Guard
Device Health Attestation Service
Device Installation
Disk NV Cache
Disk Quotas
Display
Distributed COM
Driver Installation
Early Launch Antimalware
Enhanced Storage Access
File Classification Infrastructure
File Share Shadow Copy Provider
Filesystem
Folder Redirection
Group Policy
Internet Communication Manag
iSCSI
KDC
Kerberos
2 setting(s)
Device Guard
Turn On Virtualization Based Security
Edit policy setting
Requirements:
At least Windows Server 2016, Windows 10
Description:
Specifies whether Virtualization Based Security is enabled.
Virtualization Based Security uses the Windows Hypervisor to provide support for security services. Virtualization Based Security requires Secure Boot, and can optionally be enabled with the use of DMA Protections. DMA protections require hardware support and will only be enabled on correctly configured devices.
Virtualization Based Protection of Code Integrity
Extended Standard
Setting
State
Comment
Deploy Windows Defender Application Control
Not configured
No
Turn On Virtualization Based Security
Not configured
No
Turn On Virtualization Based Security
Turn On Virtualization Based Security
Previous Setting
Next Setting
Not Configured
Enabled
Disabled
Comment:
Supported on:
At least Windows Server 2016, Windows 10
Options:
Help:
Select Platform Security Level:
Virtualization Based Protection of Code Integrity:
Require UEFI Memory Attributes Table
Credential Guard Configuration:
Secure Launch Configuration:
Specifies whether Virtualization Based Security is enabled.
Virtualization Based Security uses the Windows Hypervisor to provide support for security services. Virtualization Based Security requires Secure Boot, and can optionally be enabled with the use of DMA Protections. DMA protections require hardware support and will only be enabled on correctly configured devices.
Virtualization Based Protection of Code Integrity
This setting enables virtualization based protection of Kernel Mode Code Integrity. When this is enabled, kernel mode memory protections are enforced and the Code Integrity validation path is protected by the Virtualization Based Security feature.
The "Disabled" option turns off Virtualization Based Protection of Code Integrity remotely if it was previously turned on with the "Enabled without lock" option.
OK
Cancel
Apply

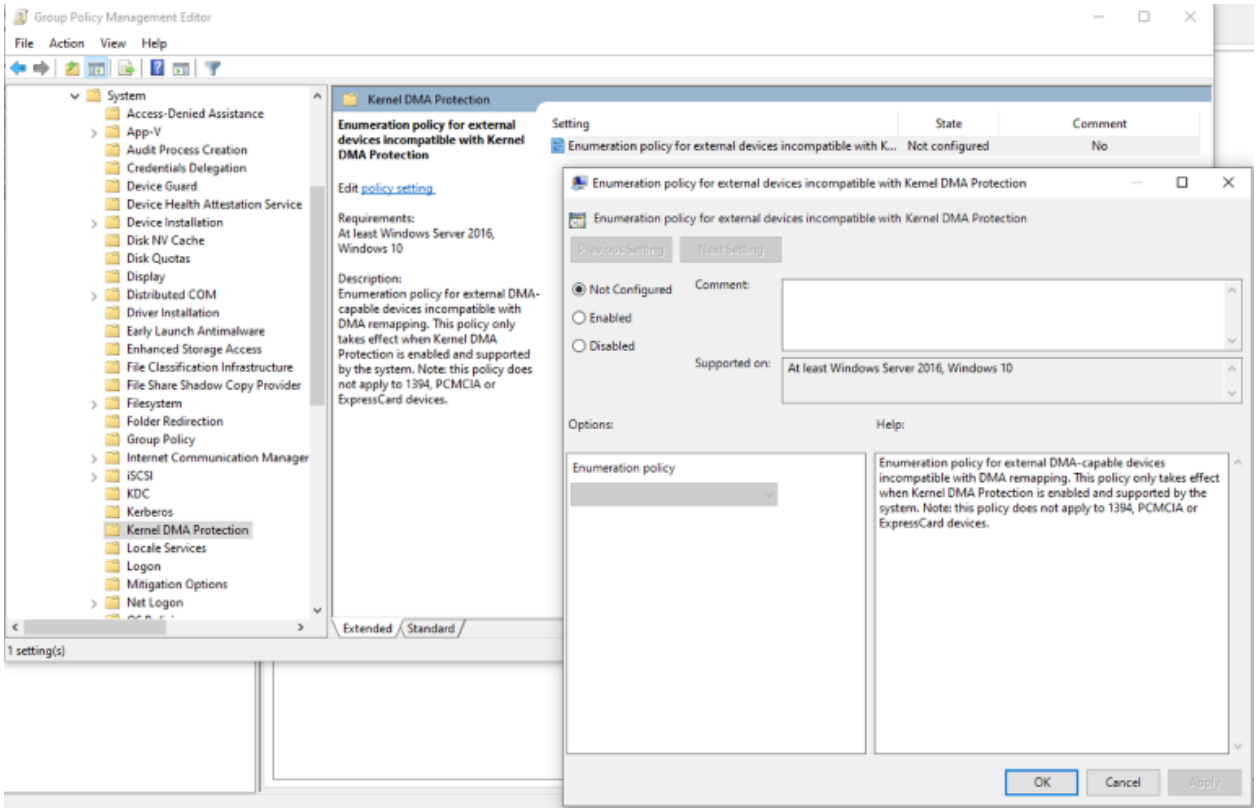
More information on the ways to enable System Guard Secure Launch are covered below:

- Mobile Device Management (MDM)
- Group Policy
- Windows Security Center
- Registry

Memory Access Protection

Windows leverages the system Input/Output Memory Management Unit (IOMMU) to block external peripherals from starting and performing DMA unless the drivers for these peripherals support memory isolation (such as DMA-remapping). Peripherals with DMA Remapping compatible drivers will be automatically enumerated, started and allowed to perform DMA remapping to their assigned memory regions.

By default, peripherals with DMA Remapping incompatible drivers will be blocked from starting and performing DMA until an authorized user signs into the system or unlocks the screen. IT administrators can ensure that this default behavior is applied to devices with DMA Remapping incompatible drivers using the DMAGuard MDM policies and selecting that the policy is enabled.



Enhanced Sign-in Security (ESS)

Windows Hello enables a user to authenticate using their biometrics or a PIN eliminating the need for a password. Biometric authentication uses facial recognition or fingerprint to prove a user's identity in a way that is secure, personal, and convenient. Enhanced Sign-in Security provides an additional level of security by leveraging specialized hardware and software components, such as Virtualization Based Security (VBS) and Trusted Platform Module to isolate and protect a user's authentication data and secure the channel by which that data is communicated.

To enable ESS in a custom image you will need to ensure that:

1. Face authentication
    a. HKLM\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\SecureBiometrics is set to 1
    b. The OEM Secure Camera driver is installed
2. Fingerprint authentication
    a. HKLM\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\SecureBiometrics is set to 1
    b. HKLM\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\SecureFingerprint is set to 1
    c. The OEM Secure Fingerprint driver is installed.

Memory Integrity (HVCI)

Memory Integrity, also known as Hypervisor-Protected Code Integrity (HVCI), is a Virtualization Based Security (VBS) feature that provides critical protection and hardening of the Windows kernel. It uses virtualization to isolate the Code Integrity (CI) decision-making function from the rest of the Windows operating system, securely managing kernel memory and the processes that run in this high-privilege environment. To enable HVCI on Windows 11 devices with supporting hardware throughout an enterprise, use any of these options:

- Windows Security app
- Microsoft Intune (or another MDM provider)
- Group Policy
- Microsoft Endpoint Configuration Manager
- Registry

Trusted Platform Module

Trusted Platform Module (TPM) technology is designed to provide hardware-based, security-related functions. A TPM chip is a secure crypto-processor that is designed to carry out cryptographic operations. The chip includes multiple physical security mechanisms to make it tamper resistant, and malicious software is unable to tamper with the security functions of the TPM. Some of the key advantages of using a TPM are:

- Generate, store, and limit the use of cryptographic keys.
- Use TPM technology for platform device authentication by using the TPM's unique RSA key.
- Help ensure platform integrity by taking and storing security measurements.

The most common TPM functions are used for system integrity measurements and for key creation and use. During the boot process of a system, the boot code that is loaded (including firmware and the operating system components) can be measured and recorded in the TPM. The integrity measurements can be used as evidence for how a system started and to make sure that a TPM-based key was used only when the correct software was used to boot the system.

Review Configuration information TPM Group Policy settings for details on how to configure TPMs via GPO. Additionally, you can utilize the TPM cmdlets to configure the TPM.

Both require that the TPM be enabled in device firmware.

Modern Device Servicing

Windows Update supports Modern Device Servicing, and the device OEM will provide Windows Update with the appropriate drivers and firmware. Enterprise administrators need only ensure that their devices can be serviced by Windows Update.

UEFI Firmware settings

When doing an OS reimage, the UEFI settings are not modified; however, in the scenario where users may have modified the UEFI settings themselves prior to a reimage, the following will assist in determining whether the settings are correct.

OEM Firmware Settings Deployment Tools

As several Windows Secured-core PC settings require enablement of specific firmware settings, an Enterprise Administrator will need to configure these either directly on the target device or by deploying firmware settings via OEM Deployment Tools.

These tools are available directly from the OEM and are outside the scope of this document. Microsoft provides information on how to perform this for Microsoft Surface device in Manage and deploy Surface driver and firmware updates.

UEFI Secure Boot

Secure boot is a security standard developed by members of the PC industry to help make sure that a device boots using only software that is trusted by the Original Equipment Manufacturer (OEM). When the PC starts, the firmware checks the signature of each piece of boot software, including UEFI firmware drivers (also known as Option ROMs), EFI applications, and the operating system. If the signatures are valid, the PC boots, and the firmware gives control to the operating system.

The OEM can use instructions from the firmware manufacturer to create Secure boot keys and to store them in the PC firmware. Enterprise Administrators will need to use their OEM firmware setting distribution tools to deploy the required Secure Boot firmware settings. For a generic description of how to enable and disable Secure Boot in firmware please review Enabling Secure Boot.

Firmware Virtualization

To support running of Virtual Machines and several Windows security services, the device must have virtualization enabled in the device firmware. The options in firmware depend on whether the device has an Intel or AMD platform. For devices using the Intel platform, you will need to enable “Intel Virtualization Technology (Intel VT)”, and for

devices using the AMD platform, you will need to enable “AMD Virtualization (AMD-V) technology”.

Windows 10 Secured-core PCs

Article • 12/17/2020

Microsoft works closely with OEM partners to help ensure that all certified Windows systems deliver a secure operating environment. Windows integrates closely with the hardware to deliver protections that take advantage of available hardware capabilities:

- Baseline Windows security – recommended baseline for all individual systems that provides foundational system integrity protections. Leverages TPM 2.0 for a hardware root of trust, secure boot and BitLocker drive encryption.
- Virtualization-based security enabled – leverages virtualization capabilities from hardware and the hypervisor to provide additional protection for critical subsystems and data.
- Secured-core – recommended for the most sensitive systems and industries like financial, healthcare, and government agencies. Builds on the previous layers and leverages advanced processor capabilities to provide protection from firmware attacks.

Secured-core PCs

Microsoft is working closely with OEM partners and silicon vendors to build Secured-core PCs that features deeply integrated hardware, firmware and software to ensure enhanced security for devices, identities and data.

Secured-core PCs provide protections that are useful against sophisticated attacks and can provide increased assurance when handling mission-critical data in some of the most data-sensitive industries, such as healthcare workers that handle medical records and other personally identifiable information (PII), commercial roles that handle high business impact and highly sensitive data, such as a financial controller with earnings data.

For general purpose laptops, tablets, 2-in-1's, mobile workstations, and desktops, Microsoft recommends using Security baselines for optimal configuration. For more info, see Windows security baselines.

Baseline Windows security is supported by Secure Boot, Bitlocker device encryption, Microsoft Defender, Windows Hello and a TPM 2.0 chip to provide a hardware root of trust for the OS platform. These features are designed to secure general purpose modern devices. If you are a decision maker purchasing new devices, your devices should meet the baseline Windows security requirements.

In addition, Windows 10 in S mode provides an additional layer of security with flexibility. S mode is a configuration that's available on all Windows editions. By ensuring only trusted applications are run on the system, S mode keeps the Windows experience fast and secured. This comes with some cost in terms of compatibility, but Intune also allows customers to install applications on an S mode system, while maintaining the S mode protections against running non-trusted applications.

What makes a Secured-core PC

Benefit	Feature	Hardware/Firmware requirement	Baseline Windows Security	Secured-core PCs
Create a hardware backed root of trust	Trusted Platform Module 2.0 (TPM)	Meet the latest Microsoft requirements for the Trusted Computing Group (TCG) specification	V	V
	Dynamic Root of Trust for Measurement (DRTM)	Enabled on device (via Secure Launch)		V
	System Management Mode (SMM)	Enabled on device (via System Guard)		V
	Secure Boot	Secure Boot is enabled in the BIOS by default.	V	V
	Memory Access Protection	The device supports Memory Access Protection (Kernel DMA Protection)		V
Ensure strong code integrity	Hypervisor Code Integrity (HVCI)	Enabled on device		V

Benefit	Feature	Hardware/Firmware requirement	Baseline Windows Security	Secured-core PCs
Provide advanced identity verification and protection	Windows Hello	If device supports Windows Hello, then those implementations must be capable of Enhanced sign-in. “Capable” means: • Designed-in SecureBIO capable components for the Windows Hello modes are supported on the device (Face and/or Fingerprint) • The device has the right SecureBIO components to enable SecureBIO functionality in a future OS release; meaning, the device BIOS implements the necessary SecureBIO SDEV table, but it is disabled by DEFAULT until supported by a future OS version.	V*	V
Protect critical data if a device is lost, stolen or confiscated	BitLocker encryption	BitLocker can leverage the TPM2.0 to encrypt and protect data”	V	V

*Possible on some devices

How to restore Secured-core PC configurations for Enterprise Administrators

Article • 06/23/2022

This document provides the steps to restore Secured-core PC configuration settings in the scenario where an Enterprise customer reimages a Secured-core PC, and subsequently needs to reenable all the Secured-core PC features. In sum, this document will cover the steps to enable the following Secured-core PC features, which can also be found on the Windows 10 Secured-core PCs webpage:

- Modern Standby
- System Management Mode (SMM) Protection
- Memory Access Protection enabled
- Enhanced Sign-in Security capable
- Memory Integrity (HVCI) enabled
- Trusted Platform Module 2.0
- Modern Device Servicing
- UEFI Secure Boot enabled
- Firmware Virtualization enabled

Managing Secured-core PC settings

Broadly, the settings for a Secured-core PC can be broken into two categories: those that can be configured via Windows, and those that are configured via the device UEFI firmware. However, even in these categories, some of the Windows settings require the device firmware to have specific settings. This document will explain how to manage these settings from these two perspectives and will call-out any firmware settings required to support those settings.

Windows settings

Windows Defender System Guard and SMM Protection

To protect critical resources such as the Windows authentication stack, single sign-on tokens, the Windows Hello biometric stack, and the Virtual Trusted Platform Module, a system's firmware and hardware must be trustworthy. Windows Defender System Guard

(WDSG) reorganizes the existing Windows 10 system integrity features under one roof and sets up the next set of investments in Windows security. It's designed to make these security guarantees:

- Protect and maintain the integrity of the system as it starts up
- Validate that system integrity has truly been maintained through local and remote attestation

Additionally, WDSG also protects against System Management Mode (SMM) attacks. SMM is a special-purpose CPU mode in x86 microcontrollers that handles power management, hardware configuration, thermal monitoring, and anything else the manufacturer deems useful. Whenever one of these system operations is requested, a non-maskable interrupt (SMI) is invoked at runtime, which executes SMM code installed by the BIOS. SMM code executes in the highest privilege level and is invisible to the OS, which makes it an attractive target for malicious activity. Even if System Guard Secure Launch is used to late launch, SMM code can potentially access hypervisor memory and change the hypervisor.

To defend against this, two techniques are used:

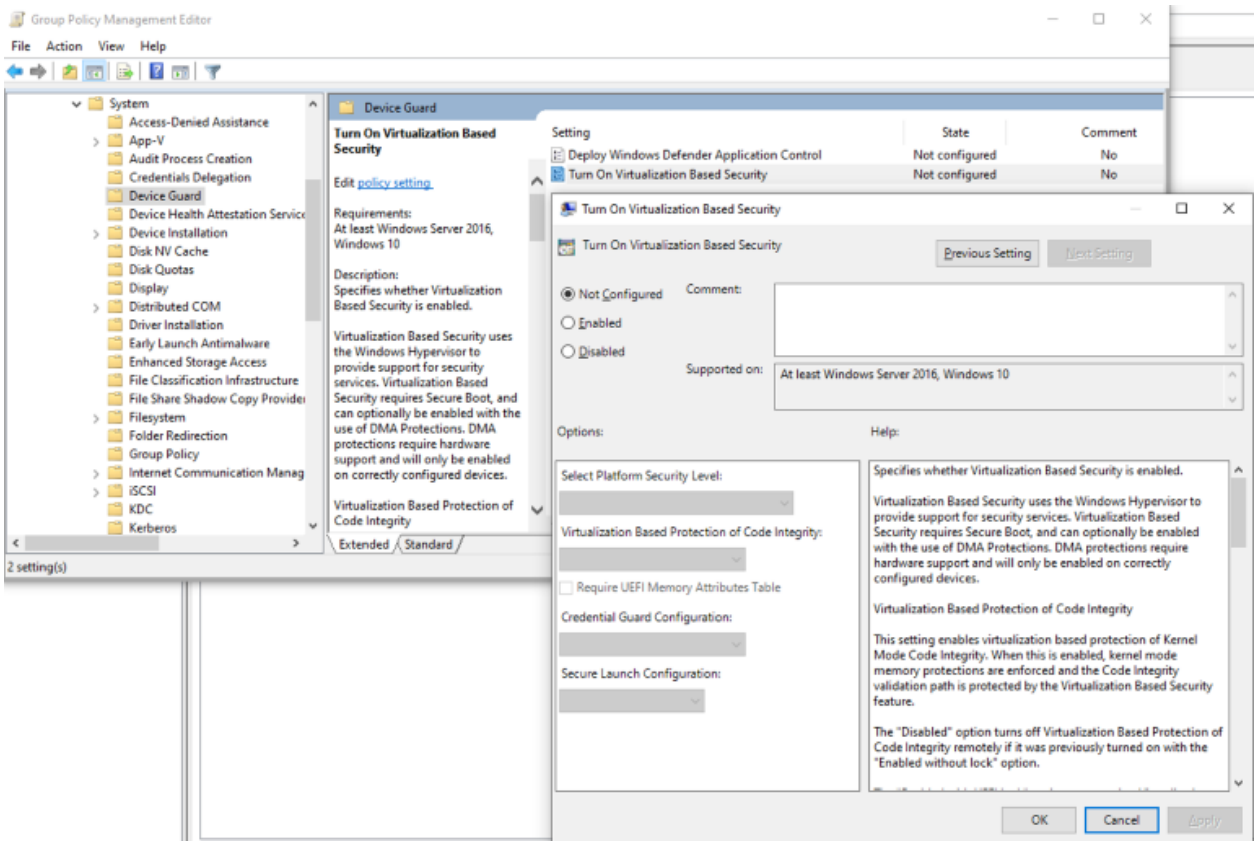
- Paging protection to prevent inappropriate access to code and data
- SMM hardware supervision and attestation

Paging protection can be implemented to lock certain code tables to be read-only to prevent tampering. This prevents access to any memory that has not been specifically assigned.

A hardware-enforced processor feature known as a supervisor SMI handler can monitor SMM and make sure it does not access any part of the address space that it is not supposed to.

SMM protection is built on top of the Secure Launch technology and requires it to function. In the future, Windows 10 will also measure this SMI Handler's behavior and attest that no OS-owned memory has been tampered with.

Reenabling WDSG protection can be done this in several ways. For example, to enable WDSG protection using GPO you would deploy the “Turn on Virtualization Based Security” template and set the Secure Launch Configuration:


Group Policy Management Editor
File Action View Help
System
Access-Denied Assistance
App-V
Audit Process Creation
Credentials Delegation
Device Guard
Device Health Attestation Service
Device Installation
Disk NV Cache
Disk Quotas
Display
Distributed COM
Driver Installation
Early Launch Antimalware
Enhanced Storage Access
File Classification Infrastructure
File Share Shadow Copy Provider
Filesystem
Folder Redirection
Group Policy
Internet Communication Manag
iSCSI
KDC
Kerberos
2 setting(s)
Device Guard
Turn On Virtualization Based Security
Edit policy setting
Requirements:
At least Windows Server 2016, Windows 10
Description:
Specifies whether Virtualization Based Security is enabled.
Virtualization Based Security uses the Windows Hypervisor to provide support for security services. Virtualization Based Security requires Secure Boot, and can optionally be enabled with the use of DMA Protections. DMA protections require hardware support and will only be enabled on correctly configured devices.
Virtualization Based Protection of Code Integrity
Extended Standard
Setting
State
Comment
Deploy Windows Defender Application Control
Not configured
No
Turn On Virtualization Based Security
Not configured
No
Turn On Virtualization Based Security
Turn On Virtualization Based Security
Previous Setting
Next Setting
Not Configured
Enabled
Disabled
Comment:
Supported on:
At least Windows Server 2016, Windows 10
Options:
Help:
Select Platform Security Level:
Virtualization Based Protection of Code Integrity:
Require UEFI Memory Attributes Table
Credential Guard Configuration:
Secure Launch Configuration:
Specifies whether Virtualization Based Security is enabled.
Virtualization Based Security uses the Windows Hypervisor to provide support for security services. Virtualization Based Security requires Secure Boot, and can optionally be enabled with the use of DMA Protections. DMA protections require hardware support and will only be enabled on correctly configured devices.
Virtualization Based Protection of Code Integrity
This setting enables virtualization based protection of Kernel Mode Code Integrity. When this is enabled, kernel mode memory protections are enforced and the Code Integrity validation path is protected by the Virtualization Based Security feature.
The "Disabled" option turns off Virtualization Based Protection of Code Integrity remotely if it was previously turned on with the "Enabled without lock" option.
OK
Cancel
Apply

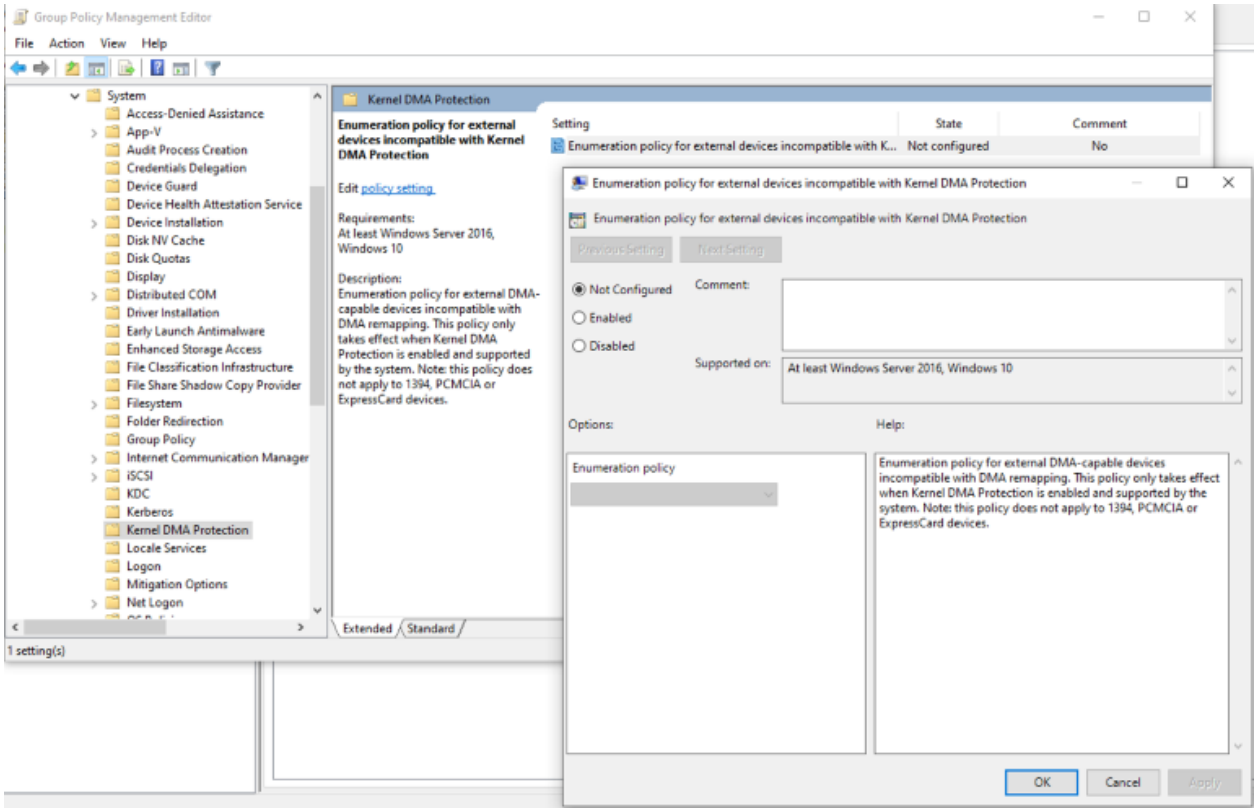
More information on the ways to enable System Guard Secure Launch are covered below:

- Mobile Device Management (MDM)
- Group Policy
- Windows Security Center
- Registry

Memory Access Protection

Windows leverages the system Input/Output Memory Management Unit (IOMMU) to block external peripherals from starting and performing DMA unless the drivers for these peripherals support memory isolation (such as DMA-remapping). Peripherals with DMA Remapping compatible drivers will be automatically enumerated, started and allowed to perform DMA remapping to their assigned memory regions.

By default, peripherals with DMA Remapping incompatible drivers will be blocked from starting and performing DMA until an authorized user signs into the system or unlocks the screen. IT administrators can ensure that this default behavior is applied to devices with DMA Remapping incompatible drivers using the DMAGuard MDM policies and selecting that the policy is enabled.



Enhanced Sign-in Security (ESS)

Windows Hello enables a user to authenticate using their biometrics or a PIN eliminating the need for a password. Biometric authentication uses facial recognition or fingerprint to prove a user's identity in a way that is secure, personal, and convenient. Enhanced Sign-in Security provides an additional level of security by leveraging specialized hardware and software components, such as Virtualization Based Security (VBS) and Trusted Platform Module to isolate and protect a user's authentication data and secure the channel by which that data is communicated.

To enable ESS in a custom image you will need to ensure that:

1. Face authentication
    a. HKLM\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\SecureBiometrics is set to 1
    b. The OEM Secure Camera driver is installed
2. Fingerprint authentication
    a. HKLM\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\SecureBiometrics is set to 1
    b. HKLM\SYSTEM\CurrentControlSet\Control\DeviceGuard\Scenarios\SecureFingerprint is set to 1
    c. The OEM Secure Fingerprint driver is installed.

Memory Integrity (HVCI)

Memory Integrity, also known as Hypervisor-Protected Code Integrity (HVCI), is a Virtualization Based Security (VBS) feature that provides critical protection and hardening of the Windows kernel. It uses virtualization to isolate the Code Integrity (CI) decision-making function from the rest of the Windows operating system, securely managing kernel memory and the processes that run in this high-privilege environment. To enable HVCI on Windows 10 devices with supporting hardware throughout an enterprise, use any of these options:

- Windows Security app
- Microsoft Intune (or another MDM provider)
- Group Policy
- Microsoft Endpoint Configuration Manager
- Registry

Trusted Platform Module

Trusted Platform Module (TPM) technology is designed to provide hardware-based, security-related functions. A TPM chip is a secure crypto-processor that is designed to carry out cryptographic operations. The chip includes multiple physical security mechanisms to make it tamper resistant, and malicious software is unable to tamper with the security functions of the TPM. Some of the key advantages of using a TPM are:

- Generate, store, and limit the use of cryptographic keys.
- Use TPM technology for platform device authentication by using the TPM's unique RSA key.
- Help ensure platform integrity by taking and storing security measurements.

The most common TPM functions are used for system integrity measurements and for key creation and use. During the boot process of a system, the boot code that is loaded (including firmware and the operating system components) can be measured and recorded in the TPM. The integrity measurements can be used as evidence for how a system started and to make sure that a TPM-based key was used only when the correct software was used to boot the system.

Review Configuration information TPM Group Policy settings for details on how to configure TPMs via GPO. Additionally, you can utilize the TPM cmdlets to configure the TPM.

Both require that the TPM be enabled in device firmware.

Modern Device Servicing

Windows Update supports Modern Device Servicing, and the device OEM will provide Windows Update with the appropriate drivers and firmware. Enterprise administrators need only ensure that their devices can be serviced by Windows Update.

UEFI Firmware settings

When doing an OS reimage, the UEFI settings are not modified; however, in the scenario where users may have modified the UEFI settings themselves prior to a reimage, the following will assist in determining whether the settings are correct.

OEM Firmware Settings Deployment Tools

As several Windows Secured-core PC settings require enablement of specific firmware settings, an Enterprise Administrator will need to configure these either directly on the target device or by deploying firmware settings via OEM Deployment Tools.

These tools are available directly from the OEM and are outside the scope of this document. Microsoft provides information on how to perform this for Microsoft Surface device in Manage and deploy Surface driver and firmware updates.

UEFI Secure Boot

Secure boot is a security standard developed by members of the PC industry to help make sure that a device boots using only software that is trusted by the Original Equipment Manufacturer (OEM). When the PC starts, the firmware checks the signature of each piece of boot software, including UEFI firmware drivers (also known as Option ROMs), EFI applications, and the operating system. If the signatures are valid, the PC boots, and the firmware gives control to the operating system.

The OEM can use instructions from the firmware manufacturer to create Secure boot keys and to store them in the PC firmware. Enterprise Administrators will need to use their OEM firmware setting distribution tools to deploy the required Secure Boot firmware settings. For a generic description of how to enable and disable Secure Boot in firmware please review Enabling Secure Boot.

Firmware Virtualization

To support running of Virtual Machines and several Windows security services, the device must have virtualization enabled in the device firmware. The options in firmware depend on whether the device has an Intel or AMD platform. For devices using the Intel platform, you will need to enable “Intel Virtualization Technology (Intel VT)”, and for

devices using the AMD platform, you will need to enable “AMD Virtualization (AMD-V) technology”.

Kernel DMA Protection (Memory Access Protection) for OEMs

Article • 11/05/2020

Kernel DMA Protection, (also known as Memory Access Protection, is a feature of a Windows 10 Secured-core PC that is supported on Intel and AMD platforms starting with Windows 10, version 1803 and Windows 10, version 1809.

With this feature, the OS and the system firmware protect the system against malicious and unintended Direct Memory Access (DMA) attacks for all DMA-capable devices:

- During the boot process.
- Against malicious DMA by devices connected to easily accessible internal/external DMA-capable ports, such as M.2 PCIe slots and Thunderbolt™3, during OS runtime.

Platform requirement	Details
64-bit CPU	Kernel DMA Protection is only supported on 64-bit IA processors with virtualization extensions, including Intel VT-X and AMD-v.
IOMMU (Intel VT-D, AMD-Vi)	All I/O devices capable of DMA must be behind an enabled (by default) IOMMU. The IOMMU is used block/unblock devices based on DMAGuard Device Enumeration Policy, and perform DMA remapping for devices with compatible drivers.
PCI Express Native Control Support	Enabling PCI Express Native Control using _OSC ACPI method is required for Kernel DMA Protection support.

Platform requirement	Details
Pre-boot DMA Protection	• System firmware must protect against pre-boot DMA attacks by implementing DMA isolation of all DMA capable devices' IO buffers pre-ExitBootServices(). • System firmware must disable the Bus Master Enable (BME) bit for all PCI root ports, that do not have children devices required to perform DMA between ExitBootServices() and the device driver being started by the OS. • At ExitBootServices(), the IOMMU must be restored by system firmware to power ON state. • No device may perform DMA outside of RMRR regions (Intel) or IVMD blocks (AMD) after ExitBootServices() until the devices' respective OS drivers are loaded and started by PnP. ◦ Performing DMA outside of RMRR regions or IVMD blocks after ExitBootServices() and prior to the device driver start by the OS will result in an IOMMU fault and potentially a system bug check (0xE6).

ACPI Kernel DMA Protection Indicators	• System firmware must set appropriate flags in ACPI tables to opt into and enable Kernel DMA Protection in OS:
	  ◦ For Intel platforms, system firmware must set "DMA_CTRL_PLATFORM_OPT_IN_FLAG" in DMAR table flags field (Intel Virtualization Technology for Direct I/O Spec, Rev 2.5 Section 8.1).
	  ◦ For AMD platforms, system firmware must set "DMA remap support" bit in the IVRS IVinfo field (AMD IOMMU Specification Rev 3.05, Section 5.2.1).
	• System firmware must tag the PCI ports that require OS runtime DMA protection with the appropriate ACPI _DSD flags:
	  ◦ Identifying externally exposed PCIe Root Ports.
	  ◦ Identifying internal PCIe ports accessible to users and requiring DMA protection.
	• For Intel platforms, DMAR tables must not include ACPI Namespace Device Descriptors (ANDD) structures.

Trusted Platform Module (TPM) 2.0	TPMs, either discrete or firmware, will suffice. For more information, see Trusted Platform Module (TPM) 2.0. • On every boot where the IOMMU (VT-D or AMD-Vi) or Kernel DMA Protection are disabled, will be disabled, or configured to a lower security state, the platform **MUST** extend an EV_EFI_ACTION event into PCR[7] before enabling DMA. • The event string SHALL be "DMA Protection Disabled". The platform firmware **MUST** log this measurement in the event log using the string "DMA Protection Disabled" for the Event Data.

Verifying Kernel DMA Protection state on a Windows 10 system

The state of Kernel DMA Protection can be verified on a given system using either of the following methods

1. Using System Information application:

    - Launch MSINFO32.exe.
    - Check "Kernel DMA Protection" field in the "System Summary" page.

2. Using Windows Security application:

    - Launch Windows Security application from the Windows Start menu.
    - Click on the “Device Security” icon.
    - Click on “Core isolation details”.
    - “Memory Access Protection” will be listed as an available Security Feature, if enabled.
        - If “Memory Access Protection” is not listed, then the feature is not enabled on the system.

Related topics

Kernel DMA Protection overview

Enabling DMA Remapping for device drivers

DMAGuard Policy

BitLocker drive encryption in Windows 11 for OEMs

Article • 05/22/2024

BitLocker drive encryption provides offline data and operating system protection by ensuring that the drive is not tampered with while the operating system is offline. BitLocker drive encryption uses a TPM, either discrete or firmware, that supports the Static Root of Trust Measurement as defined by the Trusted Computing Group.

BitLocker automatic device encryption

BitLocker automatic device encryption uses BitLocker drive encryption technology to automatically encrypt internal drives after the user completes the Out Of Box Experience (OOBE) on devices that meet hardware requirements.

> **Note**
>
> BitLocker automatic device encryption starts during Out-of-box (OOBE) experience. However, protection is enabled (armed) only after users sign in with a **Microsoft Account** or an **Azure Active Directory** account. Until that, protection is suspended and data is not protected. BitLocker automatic device encryption is not enabled with local accounts, in which case BitLocker can be manually enabled using the BitLocker Control Panel.

BitLocker automatic device encryption hardware requirements

> **Note**
>
> Starting from Windows 11 version 24H2, Microsoft has reduced hardware requirements for Automatic Device Encryption (Auto-DE) in Windows:
>
> - Auto-DE doesn't depend on Hardware Security Test Interface (HSTI) / Modern Standby anymore.
> - Auto-DE will be enabled even if untrusted Direct Memory Access (DMA) buses / interfaces are detected

This means OEMs don't need to add DMA interfaces to the AllowedBuses registry key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\DmaSecurity\AllowedBuses

These new, reduced requirements are automatically reflected in HLK tests, and no further action is required from OEMs.

BitLocker automatic device encryption is enabled when:

- The device contains a **TPM** (Trusted Platform Module), either TPM 1.2 or TPM 2.0.
- **UEFI Secure Boot** is enabled. See Secure Boot for more information.
- **Platform Secure Boot** is enabled
- Platform is **Modern Standby** or **HSTI compliant** (this requirement has been removed since Windows 11 24H2)
- There are no un-allowed **Direct memory access (DMA)** interfaces (this requirement has been removed since Windows 11 24H2)

The following tests must pass before Windows 11 will enable Automatic BitLocker device encryption. If you want to create hardware that supports this capability, you must verify that your device passes these tests.

1. **TPM**: Device must include a TPM with PCR 7 support. See System.Fundamentals.TPM20.TPM20.

    - If presence of expandable cards results in OROM UEFI drivers being loaded by UEFI BIOS during boot, then BitLocker will NOT use PCR7 binding.
    - If you are running a device that does not bind to PCR7 and Bitlocker is enabled, there are no security downsides because BitLocker is still secure when using regular UEFI PCR profile (0,2,4,11).
    - Any extra CA hash (even Windows Prod CA) before final bootmgr Windows Prod CA will prevent BitLocker from choosing to use PCR7. It does not matter if the extra hash or hashes are from UEFI CA (aka. Microsoft 3rd Party CA) or some other CA.

2. **Secure boot**: UEFI Secure Boot is enabled. See System.Fundamentals.Firmware.UEFISecureBoot.

3. **Modern Standby** requirements or **HSTI** validation. (removed since Windows 11 24H2)
   This requirement is met by one of the following:

- Modern Standby requirements are implemented. These include requirements for UEFI Secure Boot and protection from unauthorized DMA.
- Starting with Windows 10, version, 1703, this requirement can be met through HSTI test:
  a. Platform Secure Boot self-test (or additional self-tests as configured in the registry) must be reported by HSTI as implemented and passed.
  b. Excluding Thunderbolt, HSTI must report no non-allowed DMA busses.
  c. If Thunderbolt is present, HSTI must report that Thunderbolt is configured securely (security level must be SL1 – "User Authorization" or higher).

4. You must have 250MB of free space on top of everything you need to boot (and recover Windows, if you put WinRE on the system partition). For more information, see System and utility partitions.

When the requirements as listed above are met, System Information indicates the system supports BitLocker automatic device encryption. This functionality is available in Windows 10, version 1703 or after. Here's how to check System Information.

1. Click **Start**, and type **System information**
2. Right-click **System Information** app and click **Open as Administrator**. Allow the app to make changes to your device by clicking **Yes**. Some devices might require elevated permissions to view the encryption settings.
3. In **System Summary**, see **Device Encryption Support**. The value will state if the device is encrypted, or if not, reasons why it is disabled.

Un-allowed DMA capable bus/device(s) detected

This System Information status in Device Encryption Support means Windows detected at least one potential external DMA capable bus or device that may expose a DMA threat.

To resolve this issue, contact the IHV(s) to determine if this device has no external DMA ports. If confirmed by the IHVs that the bus or device only has internal DMA, then the OEM can add this to the allowed list.

To add a bus or device to the allowed list, you need to add a value to a registry key. To do this, you need to take the ownership of the **AllowedBuses** registry key first. Follow these steps:

1. Navigate to the **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\DmaSecurity\Allow**

edBuses registry key.

2. Right click the registry key and select **Permissions...**.
3. Click **Advanced**, click the **Change** link in the **Owner** field, enter your user account name, click Check Names, and then click OK three times to close all permission dialogs.
4. Right click the registry key and select **Permissions...** again.
5. Click the **Add...** button, add your user account, click Check Names, and then click OK. and then select the checkbox under Allow for Full Control. Then click OK.

Then, under the **AllowedBuses** key, add string (REG_SZ) name/value pairs for each flagged DMA capable bus that is determined to be safe:

- Key: *device friendly name /description*
- Value: PCI\VEN_*ID*&DEV_*ID*.

Ensure the IDs match the output from the HLK test. For example, if you have a safe device with a friendly name of "Contoso PCI Express Root Port", vendor ID 1022 and Device ID 157C, you would create a Registry entry named **Contoso PCI Express Root Port** as REG_SZ data type in: **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\DmaSecurity\AllowedBuses**

Where the value = "PCI\VEN_1022&DEV_157C"

Note: this registry key is ignored starting from Windows 11 24H2.

BitLocker drive encryption partition layout

BitLocker drive encryption uses a system partition separate from the Windows partition. The BitLocker system partition must meet the following requirements.

- The BitLocker system partition is configured as the active partition.
- The BitLocker system partition must not be encrypted.
- The BitLocker system partition must have at least 250 MB of free space, above and beyond any space used by required files. This additional system partition can be used to host Windows Recovery Environment (RE) and OEM tools (provided by the OEM), so long as the partition still meets the 250 MB free space requirement.

For more information see System and utility partitions, and Hard drives and partitions.

Disable BitLocker automatic device encryption

OEMs can choose to disable device encryption and instead implement their own encryption technology on a device. To disable BitLocker automatic device encryption, you can use an Unattend file and set PreventDeviceEncryption to True.

Alternately, you can update the **HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\BitLocker** registry key:

Value: **PreventDeviceEncryption** equal to True (1).

We do not recommend setting this registry key on devices with Recall feature."

Troubleshooting BitLocker HLK Tests

Recommended Prerequisites

Triage is much simpler when you know the following pieces of information about the device under test:

1. TPM spec (e.g. 1.2, 2.0)
2. BitLocker PCR profile (e.g. 7, 11 or 0, 2, 4, 11)
3. Whether the machine is non-AOAC or AOAC (e.g. Surface devices are AOAC machines)

This info is recommended but not required to perform triage.

BitLocker HLK issues are usually related to one of the following: misinterpreting test results, or PCR7 binding issues.

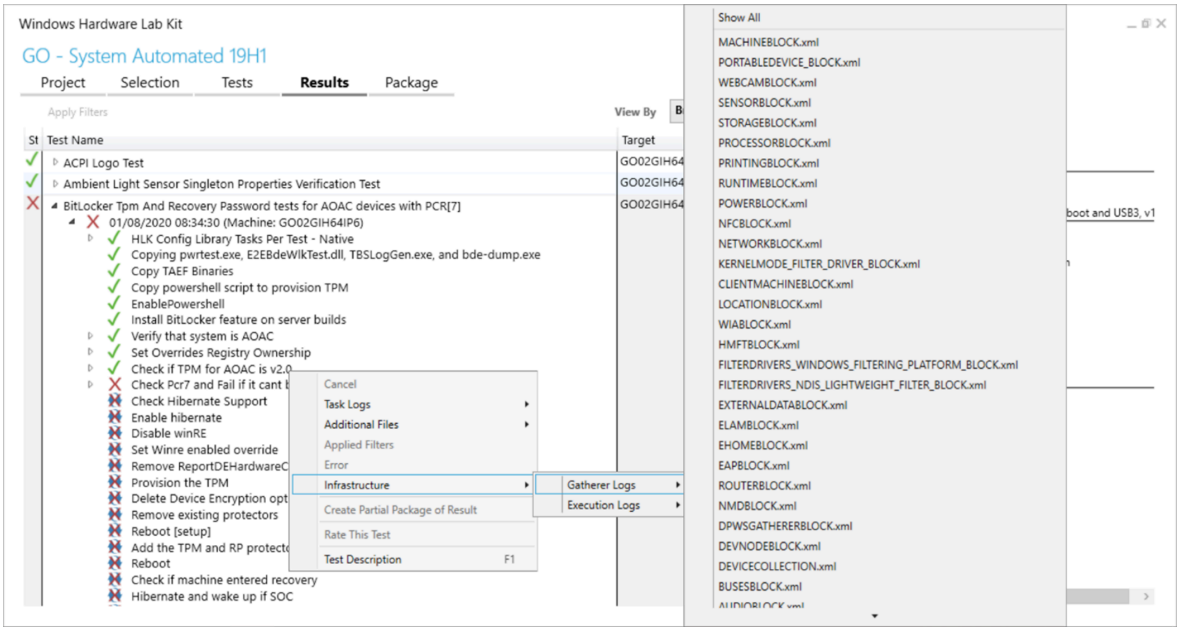
Misinterpreting test results

An HLK test consists of multiple test steps. Some test steps may fail without impacting the success/failure of the overall test. See here for more information about interpreting the results page. If some test steps have failed but the overall test passes (as indicated by a green check next to the test name), stop here. The test ran successfully and there is no more action needed on your part.

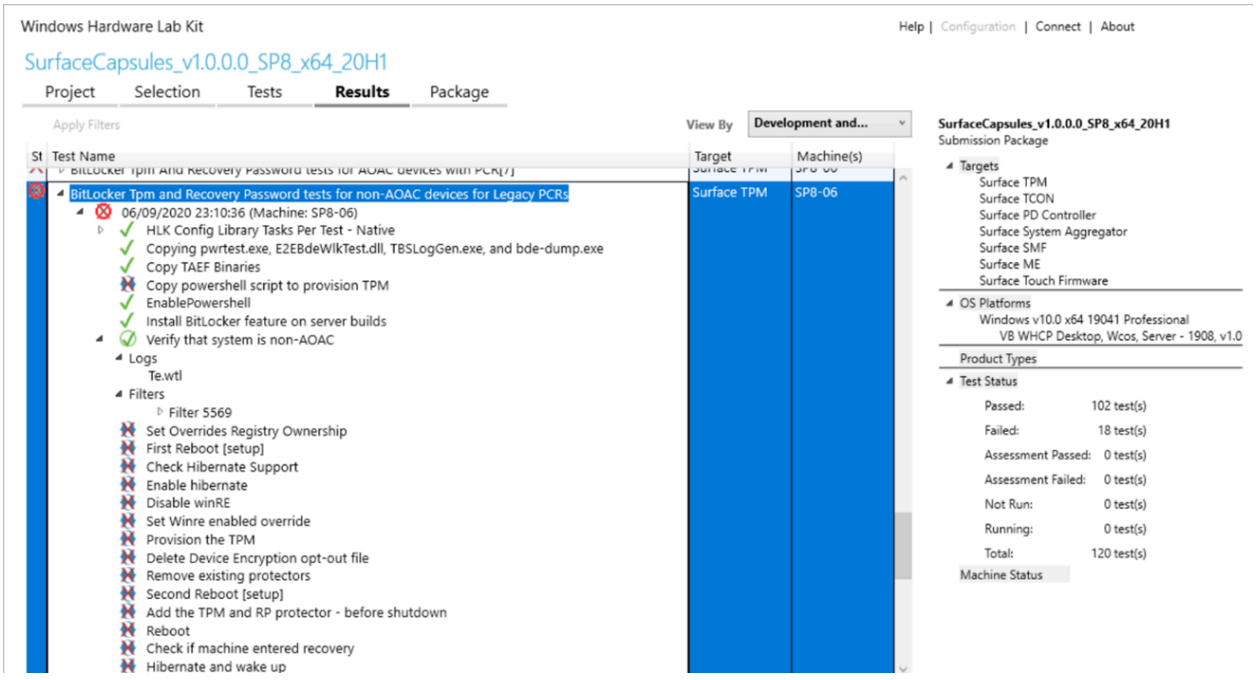
Triaging steps:

1. Confirm that you are running the right test against the machine. Right click on any step of the failing test > Infrastructure > Gatherer logs > look inside RUNTIMEBLOCK.xml for the item **IsAOAC**. If IsAOAC=true and you are running a

non-AOAC test, then disregard the failure and do not run this test against the machine. If necessary, reach out to the Microsoft Support team for an errata for passing the playlist.



2. Determine whether a filter is being applied to the test. HLK may automatically suggest a filter for an incorrectly mapped test. A filter appears as a green check mark inside a circle next to a test step. (Note that some filters may show that the subsequent test steps have failed or were canceled.) Examine the extended information about the filter by expanding the test step with the special icon. **If the filter says to disregard the test failure, stop here.**

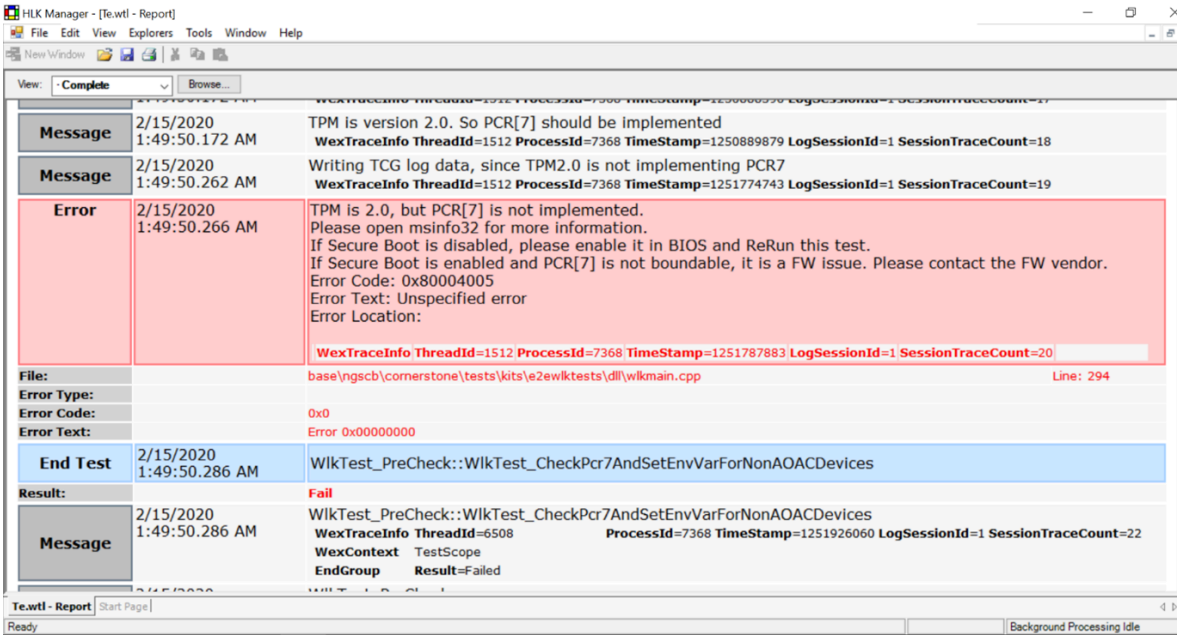


PCR7 issues

A common BitLocker issue that is specific to the two PCR7 tests is a failure to bind to PCR7.

Triaging steps:

1. Find the error message in the HLK logs. Expand the failing test step and examine the Te.wtl log. (You can also access this log by right clicking on a **test step** > **Task logs** > **Te.wtl**) Continue following the triage steps if you see this error:



2. Run msinfo32 as an administrator and check Secure Boot State/PCR7 Configuration. The test should be run with Secure Boot on. If PCR7 binding is unsupported, then run the appropriate Legacy PCR HLK test instead. If PCR7 binding is not possible, then continue following the triage steps.

3. Examine the error logs. Right click on the test task > Additional files. Usually, the PCR7 binding issue is a result of incorrect measurements into PCR7.
   a. Event logs. The Microsoft-BitLocker-Management log contains valuable error information about why PCR7 cannot be used. BitLocker HLK test should only run on a machine with BitLocker installed. Event logs shall be checked on the machine generating them.
   b. Measured boot logs. These can also be found at C:\Windows\Logs\MeasuredBoot

4. Parse the measured boot log using TBSLogGenerator.exe or equivalent. On the HLK controller, TBSLogGenerator.exe is found under the HLK tests directory where you have installed HLK, for example C:\Program Files (x86)\Windows Kits\10\Hardware Lab Kit\Tests\amd64\nttest\BASETEST\ngscb\TBSLogGenerator.exe."
   a. TBSLogGenerator.exe -lf <path to measured boot log> > OutputLog.txt

b. In OutputLog.txt, search for "PCR[07]" and examine the measurements, listed in order. The first measurement should look something like this:

```
****ID0003***0x0092-0x00e6****************************************
Event 03: EV_EFI_VARIABLE_DRIVER_CONFIG (0x80000001), 85 bytes for PCR[07]
DIGEST  = d4fdd1f14d4041494deb8fc990c45343d2277d08
PCR[07] = 3a1ea200b8fafe60c290e903c5e6443cfef67f04
EventData (53 bytes):
00000000 | 61 df e4 8b ca 93 d2 11-aa 0d 00 e0 98 03 2b 8c | aßä‹Ê“Ò ª  à˜ +Œ
00000010 | 0a 00 00 00 00 00 00 00-01 00 00 00 00 00 00 00 |
00000020 | 53 00 65 00 63 00 75 00-72 00 65 00 42 00 6f 00 | S e c u r e B o
00000030 | 6f 00 74 00 01                                  | o t
****ID0004***0x00e7-0x0467****************************************
Event 04: EV_EFI_VARIABLE_DRIVER_CONFIG (0x80000001), 897 bytes for PCR[07]
DIGEST  = b26170979280cf9b0105689b44a41fb8c5e302a5
PCR[07] = 5f599e9f15b00f4b49d3c32f58d1455cd4eb2ed2
EventData (865 bytes):
00000000 | 61 df e4 8b ca 93 d2 11-aa 0d 00 e0 98 03 2b 8c | aßä‹Ê“Ò ª  à˜ +Œ
00000010 | 02 00 00 00 00 00 00 00-3d 03 00 00 00 00 00 00 |         =
00000020 | 50 00 4b 00 a1 59 c0 a5-e4 94 a7 4a 87 b5 ab 15 | P K ¡YÀ¥ä”§J‡µ«
00000030 | 5c 2b f0 72 3d 03 00 00-00 00 00 00 21 03 00 00 | \+ðr=       !
00000040 | 55 55 55 55 55 55 55 55-55 55 55 55 55 55 55 55 | UUUUUUUUUUUUUUUU
00000050 | 30 82 03 0d 30 82 01 f5-a0 03 02 01 02 02 10 d5 | 0,  0, õ       Õ
00000060 | ee 2c 49 c6 87 1c 8d 47-19 0f 2e fe 57 20 12 30 | î,IÆ‡ G  .þW  0
00000070 | 0d 06 09 2a 86 48 86 f7-0d 01 01 0b 05 00 30 1a |    *†H†÷      0
00000080 | 31 18 30 16 06 03 55 04-03 13 0f 51 41 20 43 65 | 1 0   U    QA Ce
00000090 | 72 74 69 66 69 63 61 74-65 2e 30 1e 17 0d 31 32 | rtificate.0   12
000000a0 | 30 34 31 33 30 35 32 33-32 34 5a 17 0d 33 39 31 | 0413052324Z  391
000000b0 | 32 33 31 32 33 35 39 35-39 5a 30 1a 31 18 30 16 | 23123595 9Z0 1 0
000000c0 | 06 03 55 04 03 13 0f 51-41 20 43 65 72 74 69 66 |   U    QA Certif
000000d0 | 69 63 61 74 65 2e 30 82-01 22 30 0d 06 09 2a 86 | icate.0, "0   *†
000000e0 | 48 86 f7 0d 01 01 01 05-00 03 82 01 0f 00 30 82 | H†÷         ,   0,
```

BitLocker expects certain static root of trust measurements static root of trust measurements in PCR7, and any variation in these measurements often prohibits binding to PCR7. The following values should be measured (in order, and without extraneous measurements in between) into PCR7:

- The contents of the SecureBoot variable
- The contents of the PK variable
- The contents of the KEK variable
- The contents of the EFI_IMAGE_SECURITY_DATABASE variable (DB)
- The contents of the EFI_IMAGE_SECURITY_DATABASE1 variable (DBX)
- (optional but common EV_SEPARATOR)
- Entries in the EFI_IMAGE_SECURITY_DATABASE that are used to validate EFI Drivers or EFI Boot Applications in the boot path. BitLocker expects only one entry here.

Common issues with the measured boot log:

- UEFI debug mode on
- Missing PK or KEK variables: PK/KEK measurement has no data (e.g. 4 bytes of 0's)

- Untrusted UEFI CA signatory

Some measured boot issues, such as running with UEFI debug mode on, may be remedied by the tester. Other issues may require an errata, in which case you should reach out to the Microsoft Support team for guidance.

Applying firmware updates to devices

In addition to running HLK tests, OEMs need to test firmware updates with BitLocker turned on. To prevent devices from starting recovery unnecessarily, follow these guidelines to apply firmware updates:

1. Suspend BitLocker (required for devices bound to PCR[07] only if the firmware update changes the Secure Boot policy)
2. Apply the update
3. Restart the device
4. Resume BitLocker

The firmware update should require the device to suspend Bitlocker only for a short time, and the device should restart as soon as possible. BitLocker can be suspended programmatically just before shutting down by using the DisableKeyProtectors method in Windows Management Instrumentation (WMI).

Related topics

- Windows 10 S security features and requirements for OEMs

Secure boot

Article • 02/08/2023

Secure boot is a security standard developed by members of the PC industry to help make sure that a device boots using only software that is trusted by the Original Equipment Manufacturer (OEM). When the PC starts, the firmware checks the signature of each piece of boot software, including UEFI firmware drivers (also known as Option ROMs), EFI applications, and the operating system. If the signatures are valid, the PC boots, and the firmware gives control to the operating system.

The OEM can use instructions from the firmware manufacturer to create Secure boot keys and to store them in the PC firmware. When you add UEFI drivers, you'll also need to make sure these are signed and included in the Secure Boot database.

For information on how the secure boot process works included Trusted Boot and Measured Boot, see Secure the Windows 10 boot process.

Secure boot requirements

In order to support Secure boot, you must provide the following.

Expand table

Hardware requirement	Details
UEFI Version 2.3.1 Errata C variables	Variables must be set to **SecureBoot=1** and **SetupMode=0** with a signature database (EFI_IMAGE_SECURITY_DATABASE) necessary to boot the machine securely pre-provisioned, and including a PK that is set in a valid KEK database. For more information, search for the System.Fundamentals.Firmware.UEFISecureBoot system requirements in PDF download of the Windows Hardware Compatibility Program Specifications and Policies.
UEFI v2.3.1 Section 27	The platform must expose an interface that adheres to the profile of UEFI v2.3.1 Section 27.
UEFI signature database	The platform must come provisioned with the correct keys in the UEFI Signature database (db) to allow Windows to boot. It must also support secure authenticated updates to the databases. Storage of secure variables must be isolated from the running operating system such that they cannot be modified without detection.
Firmware signing	All firmware components must be signed using at least RSA-2048 with

Hardware requirement	Details
	SHA-256.
Boot manager	When power is turned on, the system must start executing code in the firmware and use public key cryptography as per algorithm policy to verify the signatures of all images in the boot sequence, up to and including the Windows Boot Manager.
Rollback protection	The system must protect against rollback of firmware to older versions.
EFI_HASH_PROTOCOL	The platform provides the EFI_HASH_PROTOCOL (per UEFI v2.3.1) for offloading cryptographic hash operations and the EFI_RNG_PROTOCOL (Microsoft defined) for accessing platform entropy.

Signature Databases and Keys

Before the PC is deployed, you as the OEM store the Secure Boot databases on the PC. This includes the signature database (db), revoked signatures database (dbx), and Key Enrollment Key database (KEK). These databases are stored on the firmware nonvolatile RAM (NV-RAM) at manufacturing time.

The signature database (db) and the revoked signatures database (dbx) list the signers or image hashes of UEFI applications, operating system loaders (such as the Microsoft Operating System Loader, or Boot Manager), and UEFI drivers that can be loaded on the device. The revoked list contains items that are no longer trusted and may not be loaded. If an image hash is in both databases, the revoked signatures database (dbx) takes precedent.

The Key Enrollment Key database (KEK) is a separate database of signing keys that can be used to update the signature database and revoked signatures database. Microsoft requires a specified key to be included in the KEK database so that in the future Microsoft can add new operating systems to the signature database or add known bad images to the revoked signatures database.

After these databases have been added, and after final firmware validation and testing, the OEM locks the firmware from editing, except for updates that are signed with the correct key or updates by a physically present user who is using firmware menus, and then generates a platform key (PK). The PK can be used to sign updates to the KEK or to turn off Secure Boot.

You should contact your firmware manufacturer for tools and assistance in creating these databases.

Boot sequence

1. After the PC is turned on, the signature databases are each checked against the platform key.
2. If the firmware is not trusted, the UEFI firmware must initiate OEM-specific recovery to restore trusted firmware.
3. If there is a problem with Windows Boot Manager, the firmware will attempt to boot a backup copy of Windows Boot Manager. If this also fails, the firmware must initiate OEM-specific remediation.
4. After Windows Boot Manager has started running, if there is a problem with the drivers or NTOS kernel, Windows Recovery Environment (Windows RE) is loaded so that these drivers or the kernel image can be recovered.
5. Windows loads antimalware software.
6. Windows loads other kernel drivers and initializes the user mode processes.

Related topics

- Windows 10 S security features and requirements for OEMs
- UEFI firmware requirements
- Disabling Secure Boot

Trusted Platform Module 2.0

Article • 02/08/2023

Note: Since July 28, 2016, all new device models, lines or series (or if you are updating the hardware configuration of a existing model, line or series with a major update, such as CPU, graphic cards) must implement and enable by default TPM 2.0 (details in section 3.7 of the Minimum hardware requirements page). The requirement to enable TPM 2.0 only applies to the manufacturing of new devices.

Trusted Platform Module (TPM) technology is designed to provide hardware-based, security-related functions. A TPM chip is a secure crypto-processor that helps you with actions such as generating, storing, and limiting the use of cryptographic keys. Many TPMs include multiple physical security mechanisms to make it tamper resistant, and malicious software is unable to tamper with the security functions of the TPM.

Traditionally, TPMs have been discrete chips soldered to a computer's motherboard. Such implementations allow you as the original equipment manufacturer (OEM) to evaluate and certify the TPM separate from the rest of the system. Some newer TPM implementations integrate TPM functionality into the same chipset as other platform components while still providing logical separation similar to discrete TPM chips.

TPMs are passive: they receive commands and return responses. To realize the full benefit of a TPM, you must carefully integrate system hardware and firmware with the TPM to send it commands and react to its responses. TPMs provide security and privacy benefits for system hardware, platform owners, and users.

Before it can be used for advanced scenarios, however, a TPM must be provisioned. Starting with Windows 10, the operating system automatically initializes and takes ownership of the TPM. That means that IT professionals should not have to configure or monitor the system.

For more information about the specific requirements that must be met, see System.Fundamentals.TPM20 in the Specifications and Policies download for your operating system.

IT Professionals: To understand how TPM works in your enterprise, see Trusted Platform Module.

Related topics

- Windows 10 S security features and requirements for OEMs

- Trusted Platform Module Overview
- How Windows 10 uses the Trusted Platform Module
- TPM recommendations

United Extensible Firmware Interface (UEFI) firmware requirements

Article • 02/08/2023

When the devices starts, the firmware interface controls the booting process of the PC, and then passes control to Windows or another operating system. UEFI is a replacement for the older BIOS firmware interface and the Extensible Firmware Interface (EFI) 1.10 specifications. More than 140 leading technology companies participate in the Unified EFI Forum, including AMD, AMI, Apple, Dell, HP, IBM, Insyde, Intel, Lenovo, Microsoft, and Phoenix Technologies.

UEFI benefits

Firmware that meets the UEFI 2.3.1 specifications provides the following benefits:

- Ability to support Windows 10 security features like Secure Boot, Microsoft Defender Credential Guard, and Microsoft Defender Exploit Guard. All require UEFI firmware.
- Faster boot and resume times.
- Ability to more easily support large hard drives (more than 2 terabytes) and drives with more than four partitions.
- Support for multicast deployment, which allows PC manufacturers to broadcast a PC image that can be received by multiple PCs without overwhelming the network or image server.
- Support for UEFI firmware drivers, applications, and option ROMs.

Media installation considerations

To download Windows, see the Windows 10 download page. If you want to use media like a USB flash drive, a DVD, or an ISO, dowload the Windows Media creation tool.

Boot and miminum security requirements

As the OEM, you must provide support for the features outlined in Windows Hardware Compatibility Program Specifications and Policies, specifically the following items that are divided into two groups: boot requirements and minimum security requirements.

UEFI Runtime requirements

Various versions of Windows require some or all of the following UEFI Runtime Services. It is recommended that they all be implemented for maximum compatibility:

- GetTime
- SetTime
- UpdateCapsule
- ResetSystem

Hibernation State (S4) transition requirements

Platform firmware must ensure that operating system physical memory is consistent across S4 sleep state transitions, in both size and location. Operating system physical memory is defined according to the ACPI 3.0 specification as any memory that is described by the firmware system address map interface with a memory type other than AddressRangeReserved [2], AddressRangeUnusable [5], or Undefined [any value greater than 5].

On a UEFI platform, firmware runtime memory must be consistent across S4 sleep state transitions, in both size and location. Runtime memory is defined according to the UEFI specification as any memory that is described by the GetMemoryMap() boot service, with the attribute EFI_MEMORY_RUNTIME.

Related topics

- Windows 10 S security features and requirements for OEMs

Virtualization-based Security (VBS)

Article • 03/20/2023 • Applies to: ✅ Windows 11, ✅ Windows 10

Virtualization-based security, or VBS, uses hardware virtualization and the Windows hypervisor to create an isolated virtual environment that becomes the root of trust of the OS that assumes the kernel can be compromised. Windows uses this isolated environment to host a number of security solutions, providing them with greatly increased protection from vulnerabilities in the operating system, and preventing the use of malicious exploits which attempt to defeat protections. VBS enforces restrictions to protect vital system and operating system resources, or to protect security assets such as authenticated user credentials.

One such example security solution is memory integrity, which protects and hardens Windows by running kernel mode code integrity within the isolated virtual environment of VBS. Kernel mode code integrity is the Windows process that checks all kernel mode drivers and binaries before they're started, and prevents unsigned or untrusted drivers or system files from being loaded into system memory. Memory integrity also restricts kernel memory allocations that could be used to compromise the system, ensuring that kernel memory pages are only made executable after passing code integrity checks inside the secure runtime environment, and executable pages themselves are never writable. That way, even if there are vulnerabilities like a buffer overflow that allow malware to attempt to modify memory, executable code pages cannot be modified, and modified memory cannot be made executable.

> **Note**
>
> Memory integrity is sometimes referred to as *hypervisor-protected code integrity (HVCI)* or *hypervisor enforced code integrity*, and was originally released as part of *Device Guard*. Device Guard is no longer used except to locate memory integrity and VBS settings in Group Policy or the Windows registry.

VBS requires the following components be present and properly configured.

Expand table

Hardware requirement	Details
64-bit CPU	Virtualization-based security (VBS) requires the Windows hypervisor, which is only supported on 64-bit IA processors with virtualization extensions, including Intel VT-X and AMD-v.

Hardware requirement	Details
Second Level Address Translation (SLAT)	VBS also requires that the processor's virtualization support includes Second Level Address Translation (SLAT), either Intel VT-X2 with Extended Page Tables (EPT), or AMD-v with Rapid Virtualization Indexing (RVI).
IOMMUs or SMMUs (Intel VT-D, AMD-Vi, Arm64 SMMUs)	All I/O devices capable of DMA must be behind an IOMMU or SMMU. An IOMMU can be used to enhance system resiliency against memory attacks.
Trusted Platform Module (TPM) 2.0	For more information, see Trusted Platform Module (TPM) 2.0.
Firmware support for SMM protection	System firmware must adhere to the recommendations for hardening SMM code described in the Windows SMM Security Mitigations Table (WMST) specification. The WSMT specification contains details of an ACPI table that was created for use with Windows operating systems that support VBS features. Firmware must implement the protections described in the WSMT specification, and set the corresponding protection flags as described in the specification to report compliance with these requirements to the operating system.
Unified Extensible Firmware Interface (UEFI) Memory Reporting	UEFI firmware must adhere to the following memory map reporting format and memory allocation guidelines in order for firmware to ensure compatibility with VBS. • **UEFI v2.6 Memory Attributes Table (MAT)** - To ensure compatibility with VBS, firmware must cleanly separate EFI runtime memory ranges for code and data, and report this to the operating system. Proper segregation and reporting of EFI runtime memory ranges allows VBS to apply the necessary page protections to EFI runtime services code pages within the VBS secure region. Conveying this information to the OS is accomplished using the EFI_MEMORY_ATTRIBUTES_TABLE. To implement the UEFI MAT, follow these guidelines: 1. The entire EFI runtime must be described by this table. 2. All appropriate attributes for EfiRuntimeServicesData and EfiRuntimeServicesCode pages must be marked. 3. These ranges must be aligned on page boundaries (4KB), and can not overlap. • **EFI Page Protections** -All entries must include attributes EFI_MEMORY_RO, EFI_MEMORY_XP, or both. All UEFI memory that is marked executable must be read only. Memory marked writable must not be executable. Entries may not be left with neither of the attributes set, indicating memory that is both executable and writable.

Hardware requirement	Details
Secure Memory Overwrite Request (MOR) revision 2	Secure MOR v2 is enhanced to protect the MOR lock setting using a UEFI secure variable. This helps guard against advanced memory attacks. For details, see Secure MOR implementation.
Memory integrity-compatible drivers	Ensure all system drivers have been tested and verified to be compatible with memory integrity. The Windows Driver Kit and Driver Verifier contain tests for driver compatibility with memory integrity. There are three steps to verify driver compatibility: 1. Use Driver Verifier with the **Code Integrity compatibility** checks enabled. 2. Run the Hypervisor Code Integrity Readiness Test in the Windows HLK. 3. Test the driver on a system with VBS and memory integrity enabled. This step is imperative to validate the driver's behavior with memory integrity, as static code analysis tools simply aren't capable of detecting all memory integrity violations possible at runtime.
Secure Boot	Secure Boot must be enabled on devices leveraging VBS. For more information, see Secure Boot

VBS works on VMs that have nested virtualization support. This includes all Gen2 VMs, and Gen1 VMs that support nested virtualization. A list of supported VM series is detailed in the table below.

Expand table

VM Series Name	Nested Virtualization	VM Gen
Av2	Yes	1 (certain internal sizes support gen 2)
B	No	1 and 2
Dsv2/Dv2/Dv3/Ev3	Yes	1
Dsv3/Ddsv3	Yes	1 and 2
Dsv4/Ddsv4	Yes	1 and 2
Esv3/Edsv3	Yes	1 and 2
Esv4/Edsv4	Yes	1 and 2
Ev4/Edv4	Yes	Ev4 - 1 only Edv4 -1&2

VM Series Name	Nested Virtualization	VM Gen
Dv4/Ddv4	Yes	1 and 2
Dv5/Ddv5/Dsv5/Ddsv5	Yes	1 and 2
Ev5/Edv5/Esv5/Edsv5	Yes	1 and 2
Dasv5/Dadsv5/Easv5/ Eadsv5	Yes	1 and 2
Ebsv5/Edbsv5	Yes	1 and 2
Fsv2	Yes	1 and 2
Fx	Yes	2
Lsv2	Yes	1 and 2

Related topics

For more info about Hyper-V, see Hyper-V on Windows Server 2016 or Introduction to Hyper-V on Windows 10. For more info about hypervisor, see Hypervisor Specifications.

Virtualization-based Security System Resource Protections

Article • 03/20/2023

Virtualization-based security, or VBS, uses hardware virtualization and the Windows hypervisor to create an isolated virtual environment that becomes the root of trust of the OS that assumes the kernel can be compromised. Windows uses this isolated environment to host a number of security solutions, providing them with greatly increased protection from vulnerabilities in the operating system, and preventing the use of malicious exploits which attempt to defeat protections. VBS enforces restrictions to protect vital system and operating system resources, or to protect security assets such as authenticated user credentials.

For more background on VBS please refer to Virtualization-based Security (VBS).

VBS changes the trust model

While VBS greatly improves platform security, VBS also changes the trust boundaries in a Windows PC. With VBS, the Windows hypervisor controls many aspects of the underlying hardware that provide the basis for the VBS secure environment. The hypervisor must assume the Windows kernel could become compromised by malicious code, and so must protect key system resources from being manipulated from code running in kernel mode in a manner that could compromise security assets.

Understanding processor MSRs

One area of vital system resources the hypervisor must protect from malicious use is processor model-specific registers, or MSRs. Modern processors support a great number of MSRs, many of which control key aspects of the processor's behavior. MSRs may only be read from or written to kernel mode code (that is, CPL0). Altering the settings controlled by MSRs could allow malicious kernel mode code to change the behavior of the system and allow an attacker to gain control, compromising security. In addition, many MSRs contain data about the system's operation, such as tracing or diagnostic data, which could also be used to reveal or calculate security assets. Therefore, the hypervisor must identify and protect against the misuse of any MSRs that could compromise the security posture of VBS.

MSRs are accessed via an index, which is a unique per-MSR identifier. Historically, many MSRs have been established as architectural; that is, their presence and function remain

architecturally consistent across multiple processor generations. In this case, a known MSR with a documented MSR index and definition can be relied on to control a known, published set of capabilities. However, there are also MSRs that vary across processors, and there are cases where MSR indexes have been re-purposed over time, being re-defined to refer to new sets of controls. These are very problematic for system level software, as it is difficult to encode and maintain knowledge of these controls in broadly available commercial software.

Protecting access to MSRs

In order to provide a robust security platform, MSRs must be protected from misuse from malicious kernel mode code. To enforce this, the hypervisor monitors and controls access to all MSRs. The hypervisor maintains a list of known MSR indices, and will only allow kernel mode code to access MSRs, or specific bits within MSRs, which are known to be reasonable and deemed safe. The hypervisor will block access to any MSR which is unknown to the hypervisor, or any MSR which is known through its published definition to represent a security risk. In some cases, partial access may be allowed.

When the hypervisor has blocked access to an MSR, it will log an event to the Windows System log in Event Viewer, specifying details of the attempted access.

Given the vast number of functions controlled by MSRs, it is not possible to predict the side effects of preventing MSR access to the software that initiated it. Well-written software should handle errors and failure cases gracefully, however, this is not always the case.

Reviewing MSR access events

The hypervisor will only block access to certain MSRs when VBS is enabled and running. To determine if the hypervisor has blocked access to an MSR, look in the Windows System log for event ID 12550 from the Microsoft-Windows-Hyper-V-Hypervisor. The event log entry details will contain the following information:

Id : 12550
Description : Hyper-V detected access to a restricted MSR
Details:

- Msr
- IsWrite
- MsrValue
- AccessStatus

- Pc
- ImageBase
- ImageChecksum
- ImageTimestamp
- ImageName

Supported versions of Windows

VBS is supported on all versions of Windows beginning with Windows 10 and Windows Server 2016.

Memory integrity and VBS enablement

Article • 04/07/2023

Memory integrity is a virtualization-based security (VBS) feature available in Windows 10, Windows 11, and Windows Server 2016 or higher. Memory integrity and VBS improve the threat model of Windows and provide stronger protections against malware trying to exploit the Windows kernel. VBS uses the Windows hypervisor to create an isolated virtual environment that becomes the root of trust of the OS that assumes the kernel can be compromised. Memory integrity is a critical component that protects and hardens Windows by running kernel mode code integrity within the isolated virtual environment of VBS. Memory integrity also restricts kernel memory allocations that could be used to compromise the system, ensuring that kernel memory pages are only made executable after passing code integrity checks inside the secure runtime environment, and executable pages themselves are never writable.

> **Note**
>
> Memory integrity is sometimes referred to as *hypervisor-protected code integrity (HVCI)* or *hypervisor enforced code integrity*, and was originally released as part of *Device Guard*. Device Guard is no longer used except to locate memory integrity and VBS settings in Group Policy or the Windows registry.

See Virtualization Based Security System Resource Protections for more details on these protections.

Default enablement

Memory integrity is turned on by default on clean installs of Windows 11, and previously only on clean installs of Windows 10 in S mode, on compatible hardware as described in this article. It's also turned on by default on all Secured-core PCs. On other systems that don't meet the memory integrity auto-enablement requirements, customers can opt in using any of the methods described in how to enable memory integrity. IT Pros and end users always have the final control of whether memory integrity is enabled.

Hardware features for automatic enablement

Memory integrity is turned on by default when a PC meets the following minimum hardware features:

Expand table

Component	Detail
Processor	• Intel 8th generation or later starting with Windows 11, version 22H2 (11th generation Core processors and newer only for Windows 11, version 21H2) • AMD Zen 2 architecture and newer • Qualcomm Snapdragon 8180 and newer
RAM	Minimum 8GB (Only applicable for x64 processors)
Storage	SSD with a minimum size of 64GB
Drivers	Memory integrity-compatible drivers must be installed. See Driver compatibility with memory integrity for more information about drivers.
BIOS	Virtualization must be enabled

If you're building an image that won't automatically enable memory integrity, you can still configure your image so that it's turned on by default.

> ⓘ **Note**
>
> Auto-enablement pertains only to clean installs, not upgrades of existing devices.

> ⓘ **Note**
>
> Intel 11th generation Core desktop processors are not included in current default enablement logic. However, they are a recommended platform for memory integrity and it can be enabled by the OEM.

Memory integrity and VBS controls

This section enumerates how device manufacturers and end users can interact with memory integrity and VBS. To learn about how to control memory integrity state as an administrator, see How to turn on memory integrity.

Turn on memory integrity

Although Windows will turn memory integrity on by default for most systems, there are several reasons that may prevent that from happening. As an OEM, you can ensure that memory integrity is turned on for your devices by configuring registry keys in the OS image.

Recommended configuration

Set the following two registry keys in your image to ensure memory integrity is turned on.

Expand table

Registry key	Value
HKLM\System\CurrentControlSet\Control\DeviceGuard\Scenarios\HypervisorEnforcedCodeIntegrity	Enabled=1
HKLM\System\CurrentControlSet\Control\DeviceGuard\Scenarios\HypervisorEnforcedCodeIntegrity	WasEnabledBy=1
HKLM\System\CurrentControlSet\Control\DeviceGuard\Scenarios\HypervisorEnforcedCodeIntegrity	EnabledBootId=<Current BootId>

The BootId is a counter that increments on each successful boot and can be found at the registry key: HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\PrefetchParameters\BootId The *WasEnabledBy* and *EnabledBootId* registry keys control a setting that safeguards against having an unbootable device. When set, the device will automatically turn off memory integrity if the system crashes during boot, potentially caused by memory integrity blocking an incompatible boot-critical driver. This auto-disablement feature is only available while BootId is less than EnabledBootId + 3. In some versions of Windows, the auto-disable functionality is designed to revert if the boot failures continue even after memory integrity was turned off, indicating that memory integrity was not the root cause of the failures.

> **Note**
>
> For high security systems, *WasEnabledBy* and *EnabledBootId* should NOT be set.

Troubleshooting

Identifying memory integrity state

The following volatile regkey reflects the state of memory integrity:

Expand table

Registry key	Value
HKLM\System\CurrentControlSet\Control\CI\State	HVCIEnabled

Other ways of checking memory integrity state are to look at MsInfo32 under **Virtualization-based Security Services Running** or refer to the Core isolation settings page in the Windows Security app to see the value of Memory integrity. There is also a WMI interface for checking using management tools, see Validate enabled VBS and memory integrity features.

Debugging Driver Issues

Check the Code Integrity logs to see if any drivers were blocked from loading as a result of memory integrity. These are in Event Viewer under the following path:

`Applications and Service Logs\Microsoft\Windows\CodeIntegrity\Operational`

Generally, memory integrity compatibility events have EventID=3087

Check results of memory integrity default enablement

To see details on the results of memory integrity default enablement, check the *setupact.log* and search for `HVCI`. You should see one of the following result logs, as well as the succeeding/failing checks leading to the enablement decision:

Memory integrity enabled: `SYSPRP HVCI: Enabling HVCI`

Memory integrity not enabled: `SYSPRP HVCI: OS does not meet HVCI auto-enablement requirements. Exiting now.`

If the device was opted out of memory integrity enablement via the regkey method detailed earlier, then this will be the only log from memory integrity's sysprep. If the device had a compatibility issue, it should be identified in the preceding logs with the error message:

`SYSPRP HVCI: Compatibility did not pass. VBS_COMPAT_ISSUES 0xXXXXXXXX`

The following is an enumeration of the potential VBS or memory integrity Compat Issues. Each issue is represented by a single index in a bit array, and the error message outputs the hex value resulting from each error bit being present.

Expand table

Bit Index	Compat Issue	Hex Value	Architecture
0	Unsupported architecture (eg. x86)	0x00000001	
1	SLAT required	0x00000002	x64
2	Secure Boot capability required	0x00000004	x64
3	IOMMU required	0x00000008	x64
4	MBEC/GMET Required	0x00000010	x64
5	UEFI Required	0x00000020	x64
6	UEFI WX Memory Attributes Table required	0x00000040	x64
7	ACPI WSMT table required	0x00000080	x64
8	UEFI MOR Lock required	0x00000100	x64
9	Deprecated		
10	Hardware virtualization required	0x00000400	x64
11	Secure Launch required	0x00000800	ARM64
12	Deprecated		

Bit Index	Compat Issue	Hex Value	Architecture
13	Device failing to meet 64GB minimum required volume size	0x00002000	x64, ARM64
14	System drive SSD required	0x00004000	x64, ARM64
15	Intel CET Required (Only applicable on W11 21H2)	0x00008000	x64
16	ARM SoC is not compatible with VBS	0x00010000	ARM64
17	8GB RAM required	0x00020000	x64

An example of an error code and error identification: `VBS_COMPAT_ISSUES 0x000000C0`

0x000000C0 -> 0x00000080 AND 0x00000040 -> UEFI WX Memory Attributes Table required, ACPI WSMT table required

Understanding the Windows SMM Security Mitigation Table (WSMT)

Article • 02/08/2023

The Windows SMM Security Mitigation Table (WSMT) is an ACPI table defined by Microsoft that allows system firmware to confirm to the operating system that certain security best practices have been implemented in System Management Mode (SMM) software. The WSMT table definition is described in the Windows SMM Security Mitigations Table (WMST) specification.

Background

The WSMT was defined to better support Windows Virtualization-based Security features. Please refer to Virtualization-based Security (VBS) for more background on VBS. Because SMM operates without the operating system's knowledge or control, SMM represents a significant attack surface that could be leveraged by malicious code to compromise or circumvent the OS protections enabled through VBS. Delivering a robust and secure VBS platform requires careful scrutiny and likely updates of SMM code by the OEM to eliminate common vulnerabilities that may be exploited. The WSMT contains flags which firmware can set to indicate to the operating system which of these specific security best practices have been implemented.

Implications of the WSMT on Windows VBS Support

The WSMT Protection Flags field indicates the presence of these specific SMM security mitigations in the systems firmware. Supported versions of the Windows operating system read the WSMT Protection Flags early during initialization, prior to hypervisor and VBS launch, and may elect to enable, disable, or de-feature certain security features based on the presence of these SMM Protections Flags.

Implementation Notes

Proper implementation of the security mitigations represented by the WSMT Protection Flags FIXED_COMM_BUFFERS and COMM_BUFFER_NESTED_PTR_PROTECTION will require the firmware vendor carefully evaluate and possibly re-design the System Management Interrupt (SMI) handlers. All SMI handlers must be restricted to only access

(read or write) to allowable memory regions that contain MMIO and EFI-allocated memory. It is not sufficient to check that pointers in SMM do not reference memory entirely outside of SMM. Rather, all SMM pointers must be validated to be within these safe memory regions. This prevents SMM from being exploited in a "confused deputy" attack, which could then be leveraged to compromise Windows VBS features. The above-mentioned Protection Flags refer only to input validation and pointer checks and do NOT currently require enforcement via SMM page protections. For example, SMM should not read or write to memory that was described by the firmware as EfiConventionalMemory because it may contain secrets or cause software to behave unpredictably.

Validating the WSMT Protections

Because SMM is opaque to the operating system, it is not possible to produce a test which runs in Windows to verify that the protections prescribed in the WSMT specification are actually implemented in SMM. From the operating system, the only check that is possible is to look for the presence of the WSMT, and check the state of all defined Protection Flags.

Therefore, it is the responsibility of the OEM to carefully scrutinize each system's SMM code and ensure that firmware complies with the guidance outlined in the WSMT specification and this article. No Protection Flag should be set to "true" until the OEM has confirmed that the mitigations corresponding to each Protection Flag value have been properly implemented. Failing to adhere to this as a best practice will leave the platform vulnerable to compromise, and negate the effectiveness of multiple OS protections and Windows security features which rely on VBS to maintain robust security boundaries.

Supported versions of Windows

Support for the WSMT is included in the following versions of Windows:

- Windows Server Technical Preview 2016
- Windows 10, version 1607
- Windows 10, version 1703
- Windows 10, version 1709

Microsoft Defender Application Guard hardware requirements

Article • 02/08/2023

Application Guard helps to isolate enterprise-defined untrusted sites, protecting an enterprise while its employees browse the Internet. If an employee goes to an untrusted site through either Microsoft Edge or Internet Explorer, Microsoft Edge opens the site in an isolated Hyper-V-enabled container, which is separate from the host operating system. This container isolation means that if the untrusted site turns out to be malicious, the host PC is protected, and the attacker can't get to the enterprise data.

As an OEM, you provide the hardware necessary to enable Application Guard. Here are the requirements.

Expand table

Requirement	Details
64-bit CPU	A 64-bit computer is required for hypervisor and virtualization-based security (VBS). For more information about VBS, see Virtualization Based Security (VBS).
CPU virtualization extensions	Extended page tables, also called Second Level Address Translation (SLAT) and one of the following virtualization extensions for VBS: VT-x (Intel) **-OR-** AMD-V
Memory	Microsoft requires 8GB for optimal performance
Hard drive	5 GB free space, solid state disk (SSD) recommended
IOMMU Support	Not required but highly recommended

Related topics

- Windows 10 S security features and requirements for OEMs
- HVCI hardware requirements
- Microsoft Defender Credential Guard hardware requirements

Microsoft Defender Credential Guard hardware requirements

Article • 02/08/2023

Microsoft Defender Credential Guard uses virtualization-based security to isolate and protect secrets (e.g., NTLM password hashes and Kerberos ticket-granting tickets) to block pass-the-hash or pass-the-ticket (PtH) attacks. When Microsoft Defender Credential Guard is enabled, NTLMv1, MS-CHAPv2, Digest, and CredSSP cannot use the signed-in credentials. Thus, single sign-on does not work with these protocols. However, applications can prompt for credentials or use credentials stored in the Windows Vault which are not protected by Microsoft Defender Credential Guard with any of these protocols.

It is strongly recommended that valuable credentials, such as the sign-in credentials, not be used with any of these protocols. If these protocols must be used by domain or Azure AD users, secondary credentials should be provisioned for these use cases.

When Microsoft Defender Credential Guard is enabled, Kerberos does not allow unconstrained Kerberos delegation or DES encryption, not only for signed-in credentials, but also prompted or saved credentials.

Note: Beginning with Windows 10 version 1709 and Windows Server version 1709, when Intel TXT or SGX are enabled in a platform via the BIOS, Hypervisor-Protected Code Integrity (HCVI) and Credential Guard are not impacted and will function as expected. HVCI and Credential Guard are not supported on earlier versions of Windows when Intel TXT or SGX are enabled in a platform via the BIOS.

For a better understanding of what Microsoft Defender Credential Guard is and what attacks it protects againt, see Deep Dive into Credential Guard.

IT Professionals: To learn how to deploy Microsoft Defender Credential Guard in your enterprise, see Protect derived domain credentials with Credential Guard.

For a device to support Microsoft Defender Credential Guard as specified in the Windows Hardware Compatibility Requirements (WHCR), you as the OEM must provide the following hardware, software, or firmware features.

Expand table

Requirement	Details
Secure Boot	Hardware-based Secure Boot must be supported. To learn more, see Secure Boot.
Secure Boot configuration and management	• You must be able to add ISV, OEM, or Enterprise Certificate to the Secure Boot database at manufacturing time. • Microsoft UEFI CA must be removed from the Secure Boot database. Support for 3rd-party UEFI modules is permitted but should leverage ISV-provided certificates or OEM certificate for the specific UEFI software.
Secure firmware update process	Like UEFI software, UEFI firmware can have security vulnerabilities. It is essential to have the capability to immediately patch such vulnerabilities when found through firmware updates. UEFI firmware must support secure firmware update following Hardware Compatibility Specification for Systems for Windows 10 under System.Fundamentals.Firmware.UEFISecureBoot.
United Extensible Firmware Interface (UEFI)	To lern more, see United Extensible Firmware Interface (UEFI) firmware requirements.
Virtualization-based security (VBS)	Hypervisor-Protected Code Integrity requires VBS. You can learn more about VBS by reading Virtualization-based Security (VBS).

Hypervisor-Protected Code Integrity and Credential Guard Readiness Tool

To determine if a device is able to run HVCI and Credential Guard, download the HVCI and Credential Guard hardware readiness tool.

Related topics

- Windows 10 S security features and requirements for OEMs
- Virtualization-based Security (VBS)
- Microsoft Defender Application Guard hardware requirements

Factory Encrypted Drives

Article • 02/08/2023

You can install Windows on factory-encrypted drives, also known as encrypted hard disk drives (eHDD). A *factory-encrypted drive* is a drive that is capable of full-disk encryption.

By default, when you install Windows on a factory-encrypted drive, Windows automatically encrypts the drive by using Trusted Computing Group (TCG) and IEEE 1667 transport encryption standards.

Requirements

To install Windows onto a factory-encrypted drive, use the following:

1. Firmware: UEFI version 2.3.1 that has been configured to use the EFI storage security protocol.
2. Hardware: a hard disk drive that is capable of using TCG and IEEE 1667 transport encryption standards.

Using other encryption standards

To use another encryption standard on your drive, you must first disable the automatic drive provisioning that Windows provides. To do this on a new installation, set the Microsoft-Windows-EnhancedStorage-Adm/TCGSecurityActivationDisabled Unattend setting to **true**.

Related topics

- Windows 10 S security features and requirements for OEMs
- Hard Drives and Partitions

Adaptive brightness

Article • 10/21/2022

Adaptive brightness is the screen brightness set automatically by the system in response to the ambient light sensor reading. Adaptive brightness provides users a more responsive display experience where brightness adapts automatically to the user's environment.

New in Windows 11

- The implementation for auto-brightness has been radically simplified for OEMs wishing to integrate light sensors into Windows 11 using a new paradigm, a bucketized ambient light response (ALR) curve. For more information, see Ambient light response curve changes for Windows 11.
- The new method does not rely on the previous OEM-configurable lux-to-brightness curve. The default curve is more stable and reliable, plus easier to integrate.
- Settings pages have new visuals. Updates to the Display settings page are noted in this article. There is a user toggle for Content Adaptive Brightness Control (CABC).

The following improvements in Windows 10 version 1903 (19H1) are still in place:

- Automatic brightness is enabled by default
- Users can control brightness using the action center slider
- Adaptive brightness registry configuration parameters

Optimizing display brightness steps and transitions

The number of brightness levels that a display device exposes is important. Two approaches are possible:

1. Percentage-based: Controlling brightness using percentage values, supporting 101 levels (zero through 100) of backlight control.
2. Nits-based (recommended): Controlling brightness using nits values allows for a fined-grained control of backlight levels. And therefore, enables very smooth and accurate brightness transitions.

Windows detects the type of brightness interface exposed by the display driver and selects the most appropriate one. If a display driver only exposes the DXGK_BRIGHTNESS_INTERFACE_2 interface, the system will control brightness using percentage values. If a display driver exposes the DXGK_BRIGHTNESS_INTERFACE_3 interface, Windows 10, version 1809 and later controls brightness using nits values. The DXGK_BRIGHTNESS_INTERFACE_3 interface is ignored on down-level Windows versions. If both DXGK_BRIGHTNESS_INTERFACE_2 and DXGK_BRIGHTNESS_INTERFACE_3 interfaces are exposed by the display driver, Windows 10, version 1809 and later will control brightness using nits values. Down-level Windows versions will control brightness using percentage values.

Brightness and display considerations

If the system supports setting brightness in nits by having the display drive expose the DXGK_BRIGHTNESS_INTERFACE_3 interface, the display should be properly calibrated. Calibration should be performed at various intensities by measuring nits values with a high-quality nits meter at different locations

of the display while displaying a white background. Tools to measure display brightness are called luminance meters or brightness meters, and are available for purchase from electronic equipment vendors and online retailers.

The display implementation must be carefully optimized. Specifically:

- Ensure that display is capable of dimming smoothly across all the accessible brightness levels.
- Sufficient display brightness levels should be exposed to ensure smooth dimming. At least 101 levels are recommended.

Controlling brightness using nits values

Starting with Windows 10, version 1809, the system will control brightness using nits on devices which display driver exposes the DXGK_BRIGHTNESS_INTERFACE_3 interface. The nit (candela per square meter) is an International System of Units (SI) unit of luminance. When a device bears a properly calibrated display and properly calibrated sensor, brightness control should work out of the box. There is no need for any ALR curve on these devices.

For these systems, it is critical that light sensors and display are accurately calibrated. Windows 10, version 1809 tolerates small inaccuracies that may occur during the manufacturing process of these systems. Placement of the components, glass transparency, and similar factors can greatly influence the lux and nits. Therefore, calibration on nits-based brightness systems should be done at least once with the final form factor design as opposed to doing calibration on development form factors and applying the result to the final form factor.

Per-device calibration during production of each device gives the best end results.

Controlling brightness using percentage values

Systems that don't support nits brightness control must support percentage values. On percentage systems, a mapping between the backlight percentages and luminance values is necessary. The mapping of backlight percentages to luminance values should follow an exponential pattern. On nits-based brightness systems, since the each nit level is expected to be calibrated, an inbox percentage to nits mapping is provided. This inbox percentage to nits mapping uses human vision and color science research to provide a perceptually linear brightness slider. As long as the nit levels are calibrated correctly as described in DXGK_BRIGHTNESS_INTERFACE_3, the perceptual luminance difference between zero percent and 1% will be automatically equivalent to the perceptual luminance difference between 1% and 2%, and so forth.

Human vision is more sensitive to small changes in screen brightness output at low light levels, thus more backlight levels should be allocated to the lower brightness range to accommodate smoother transitions. The difference between 1% and 2% in nits should be smaller than the difference from 10% and 11%, for example. This means that 50% of the screen's maximum luminance will not be mapped to the 50% backlight level.

For an acceptable user experience, the lowest level of brightness (0%) must result in a low, but readable display. Users can get into inextricable situations when setting brightness to 0% on devices that map that value to 0 nits, as the controls to brighten the screen are not visible anymore. The screen must be bright enough at 0% for the user to interact with the UI on the display. On devices supporting the DXGK_BRIGHTNESS_INTERFACE_3 interface, 0% is automatically restricted to a minimum of 5 nits by Windows.

Ambient light response curve changes for Windows 11

Some concerns were raised regarding the Windows 10 implementation of adaptive brightness:

- Poor ambient light sensor readings, especially in very dark or very bright environments
- Display panels that are unable to adapt to every percentage or nits-value
- Difficulty finding the optimal lux to nits mapping

These concerns result in:

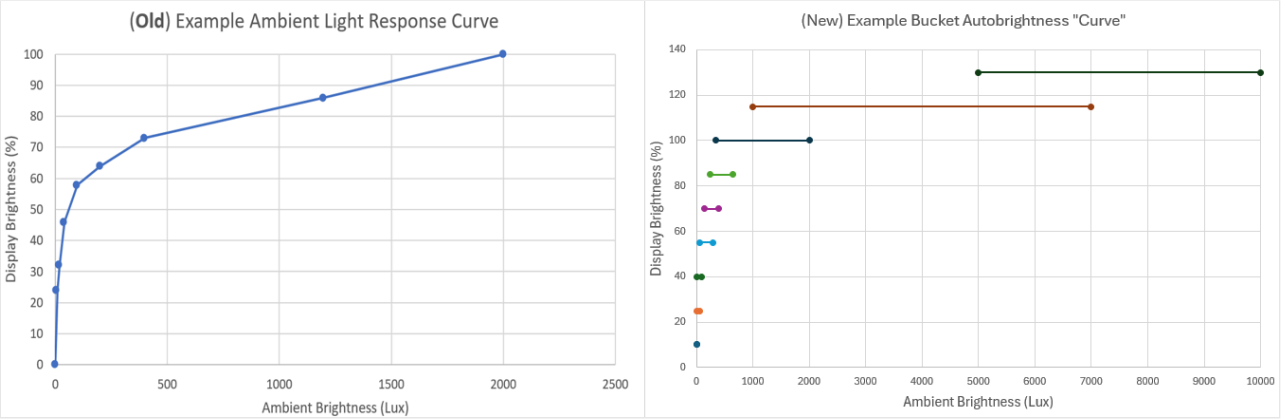
- Continual fluctuations in display brightness due to inaccuracy of the ambient light sensor (ALS)
- Slight changes in nits prominent in low lighting environments

To address these concerns, the ambient light response was changed to use a bucketized curve. The default value mappings between lux buckets and screen brightness percentages are shown in the table below. OEMs can opt-out of the bucketized curve through customization options and use the older curve.

Expand table

Bucket	Min Lux	Max Lux	Percentage
1	0	10	10
2	5	50	25
3	15	100	40
4	60	300	55
5	150	400	70
6	250	650	85
7	350	2000	100
8	1000	7000	115
9	5000	10000	130

The following image compares changes to the default ALR curve from Windows 10 to Windows 11. An example of a default curve is on the left, and a bucketized default curve for Windows 11 is on the right. The curves illustrated below are only examples of curves that could be set as defaults. The actual default curve for a device depends on a number of factors and can vary depending on the device manufacturer.



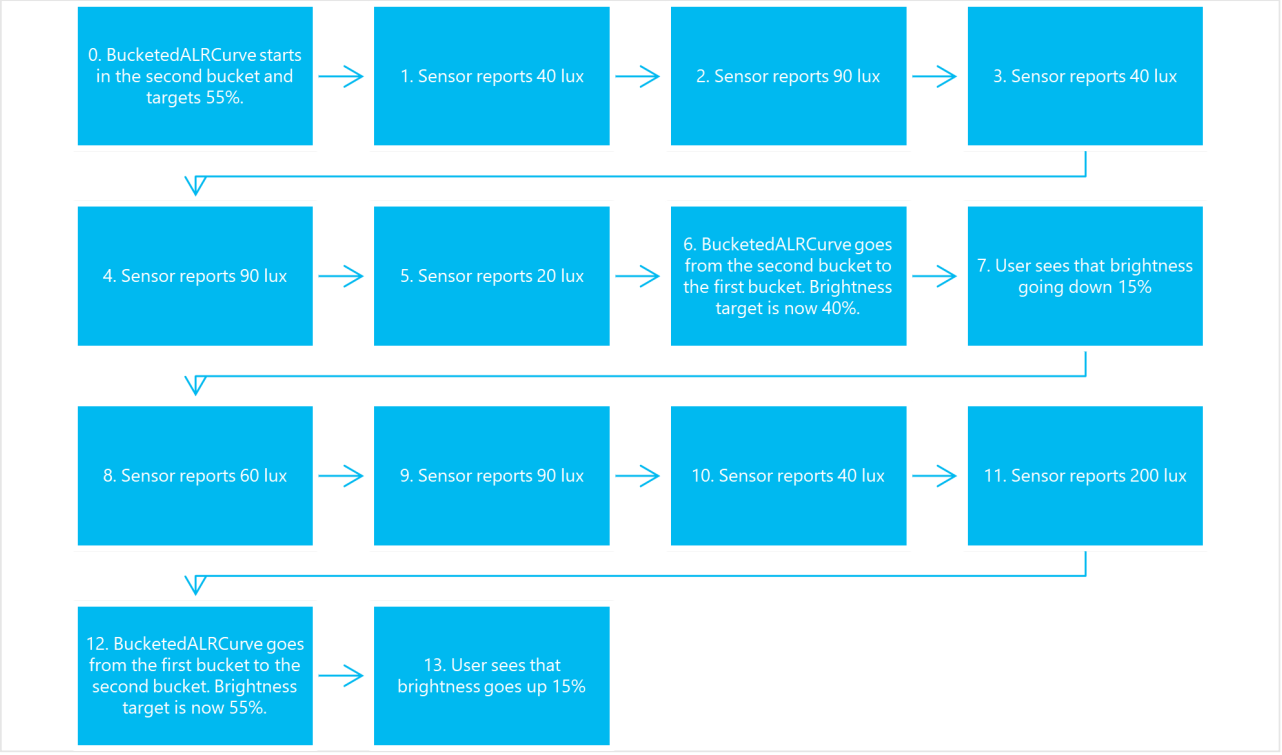
> ⓘ Note
>
> **Systems upgrading from Windows 10 to Windows 11**: No changes to hardware or firmware outside of existing Windows 10 adaptive brightness requirements, these are internal changes to Windows 11. Existing Windows 10 adaptive brightness capable systems will experience the new bucketized curve when updated to Windows 11.

Bucketed auto brightness functional overview

To reduce frequent display brightness fluctuations in response to fluctuating readings from the ALS, we introduced bucketed auto brightness. A range of lux values are mapped to a single target brightness percentage value. The display brightness is then transitioned to the target percentage value. On DXGK_BRIGHTNESS_INTERFACE_2 interface-based devices, the target percentage is used as it is. On DXGK_BRIGHTNESS_INTERFACE_3 interface-based devices, the target percentage is converted into the corresponding nits value and used. Based on our experiments under various lighting conditions, the lux ranges are partitioned into seven different overlapping buckets and assigned a corresponding display brightness target. A reading from the ALS is mapped to a target brightness using this look up table. The buckets introduce the necessary hysteresis effect that prevents fluctuations in the display brightness for non-significant changes in ALS readings. Overlapping buckets help with smooth transitions between the buckets when the ALS reading changes widely.

Bucketed auto brightness always starts with the second bucket, 55% target display brightness, as that is the most common lighting condition users experience. As the brightness changes, the target transition moves to the corresponding higher or lower bucket. The slider is animated in response to the bucket transitions.

Here is a sample sequence of events with the inbox bucketed ALR curve:


0. BucketedALRCurve starts in the second bucket and targets 55%.
1. Sensor reports 40 lux
2. Sensor reports 90 lux
3. Sensor reports 40 lux
4. Sensor reports 90 lux
5. Sensor reports 20 lux
6. BucketedALRCurve goes from the second bucket to the first bucket. Brightness target is now 40%.
7. User sees that brightness going down 15%
8. Sensor reports 60 lux
9. Sensor reports 90 lux
10. Sensor reports 40 lux
11. Sensor reports 200 lux
12. BucketedALRCurve goes from the first bucket to the second bucket. Brightness target is now 55%.
13. User sees that brightness goes up 15%

For this sequence, the brightness only went up and down twice overall even though the ambient light fluctuated frequently.

The device starts in the second bucket. When the sensor fluctuates from 40-90 lux, the brightness percent never changes since 40 and 90 lux both belong in the second bucket.

When the sensor reports a sample of 20 lux, the device goes to the first bucket as 20 lux is no longer in the second bucket. The brightness is then stable for a couple of lux values. Note that the 90 and 40 lux values do not move the brightness back to the second bucket as these values exist in the first bucket.

When the sensor reports a sample of 200 lux, the device goes to the second bucket as 200 lux exceeds the maximum lux value of the first bucket.

Outdoor scenarios

Some nits devices support boost ranges. This means that the brightness can exceed 100% if auto brightness allows it. By setting the sixth and seventh buckets past 100%, these buckets will be triggered when the users go into direct sunlight on devices that support boost ranges.

Extremely dark scenarios

Many sensors do not handle very dark scenarios well, sending fluctuating ALS values. Because the lowest bucket goes up to 100 lux, brightness should not change frequently in these dark environments.

Auto brightness registry keys

OEM customizations mentioned in this section are with respect to the following registry key in Windows 11:

`Computer\HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\AdaptiveDisplayBrightness\{23B44AF2-78CE-4943-81DF-89817E8D23FD}`

Expand table

Key	Format	Usage
AutobrightnessLuxToNitsCurve	REG_SZ	The LUT for ALS lux reading to target nits curve. Example: "1:8,2:25,5:35,10:60,20:90,40:90,100:130,400:170,700:200,2000:400,3000:500,4500:700"

Brightness hysteresis for custom ALR curve in Windows 11

Most systems will leverage a bucketized auto brightness curve by default in Windows 11. Hysteresis and transitions will not be used in the default bucketized curve. The following parameters should be used when an OEM opts-in to using a customized ALR curve.

Expand table

Key	Format	Usage
UpperBrightnessHysteresisLut	REG_SZ	The LUT for the upper bound brightness hysteresis threshold. Example: "10000:50000,20000:40000,50000:10000" represents a LUT defined by (input 10000millinits, hysteresis 50000millinits), (20000, 40000), and (50000, 100000) in that specific order
LowerBrightnessHysteresisLut	REG_SZ	The LUT for the lower bound brightness hysteresis threshold.

Brightness transitions for custom ALR curve in Windows 11

Expand table

Key	Format	Usage
MinBrightnessTransitionNitDelta	REG_DWORD	Minimum nit delta needed to trigger a brightness transition defined in millinits
DefaultBrightnessTransitionInterval	REG_DWORD	Default brightness transition internal time between two transition points defined in milliseconds
MinBrightnessTransitionInterval	REG_DWORD	Minimum brightness transition internal time between two transition points defined in milliseconds
MaxBrightnessTransitionInterval	REG_DWORD	Maximum brightness transition internal time between two transition points defined in milliseconds

Test cases

This section discusses testing the ambient light sensor.

ALS calibration

Ensure the ambient light sensor is properly calibrated for a given set of ambient lights. Verify that the sensor lux readings are accurate.

ALS calibration applies to

All systems supporting adaptive brightness

ALS calibration setup and tools

- A controllable (dimmable) source of light capable of generating different lux levels
- A light meter measuring light in lux
- MonitorBrightnessApp or SensorExplorer to visualize the values reported by the ambient light sensor

ALS calibration test procedure

1. Set the light meter next to the device. The light meter should be as close to the ambient light sensor as possible without negatively interacting with it.
2. Start the MonitorBrightnessApp
3. In a dark room, use the source of light to change the level of ambient light to different lux levels
4. Read the lux meter and the value reported by the MonitorBrightnessApp. The values should be identical

ALS calibration test variations

1. Use different types of controllable lights, such as incandescent lights, CFL, and LED

2. Use different angles

ALS calibration evaluation (pass or fail)

The lux values reported by the MonitorBrightnessApp should be identical to the values reported by the light meter.

ALS calibration triage notes

Work with your sensor hardware manufacturer to understand how to calibrate the sensor.

ALS granularity

Make sure the ambient light sensor changes are fine grained, with no delay

ALS granularity applies to

All systems supporting adaptive brightness

ALS granularity setup and tools

- Use a light source with a controllable dimmer to finely ramp the ambient light up and down. The light source should be able to smoothly ramp the light up and down.
- Use the MonitorBrightnessApp in the BrightnessTests folder to visualize the ambient light sensor response.

ALS granularity test procedure

1. In a dark room, use the dimmer to smoothly ramp the light level up and down
2. Use the MonitorBrightnessApp to visualize the ambient light sensor response. The response should match the changes applied to the dimmer.

ALS granularity evaluation (pass or fail)

The ambient light sensor response should closely match the changes applied to the dimmer. A linear change to the dimmer should result in a linear response from the ambient light sensor. The ALS response should *not* be discreet. A change to the dimmer should be immediately visible in the MonitorBrightnessApp with no delay.

ALS granularity triage notes

Work with your sensor hardware manufacturer to understand how the ambient light sensor transitions can be smoothen out and how delays can be reduced.

Sleep transitions

Make sure the ambient light sensor is still functional when coming out of sleep or when the lid is opened.

Sleep transitions applies to

All systems supporting adaptive brightness

Sleep transitions setup and tools

- Use a light source to switch the ambient light on and off.
- Use the MonitorBrightnessApp in the BrightnessTests folder to visualize the ambient light sensor response.

Sleep transitions test procedure

1. In a dark room, switch the device on or open the lid.
2. Use the MonitorBrightnessApp to visualize the ambient light sensor reading. Make sure the sensor reads a low lux value.
3. Keep the MonitorBrightnessApp running throughout the following steps
4. Close the lid or switch the device off
5. Switch the light on, make sure the ambient light is bright
6. Open the lid or switch the device on
7. Connect to the desktop and observe the value in the MonitorBrightnessApp. The lux value should immediately reflect the actual ambient light.

Sleep transitions evaluation (pass or fail)

The ambient light sensor should send a sample reading when exiting connected standby or when the lid is opened.

Sleep transitions triage notes

Work with your sensor hardware manufacturer to understand how the ambient light sensor can be fixed.

Smooth transitions

Make sure the display panel smoothly changes brightness.

Smooth transitions applies to

All systems supporting adaptive brightness

Smooth transitions setup and tools

- Use the BrightToDim.ps1 script, found in the BrightnessTests folder, to linearly ramp the brightness up and down. Switch your device to manual brightness. The script will ramp the screen brightness down from 100 to 0.
- Optional: use a brightness (nits) meter to measure the screen brightness

Smooth transitions test procedure

1. Make sure the screen is displaying as much white as possible For example, open Notepad and maximize it on the screen.
2. Start the BrightToDim.ps1 script and observe the screen behavior. The screen should transition as smoothly as possible, without any jumps in brightness.

Smooth transitions evaluation (pass or fail)

The screen brightness should smoothly transition up and down, there should not be any visible jumps in brightness.

Smooth transitions triage notes

Work with your hardware manufacturer to understand how display panel brightness changes can be made smooth.

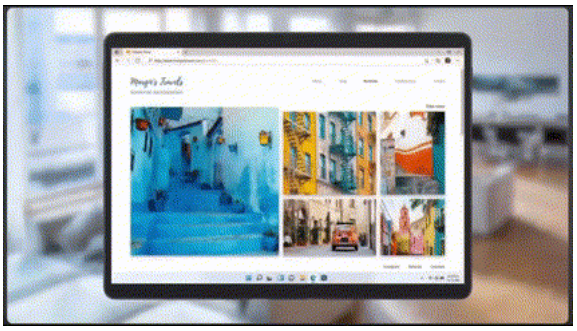
See also

- Sensor Device Driver Reference
- Ambient Light Sensor Component Guidelines
- Overview of Universal Sensor Driver Model
- Windows sensor driver samples

Adaptive Color

Article • 11/25/2024

The majority of displays today have a white point of 6500 K. However, most usage environments do not match this white point. Common office lighting is around 5000 K, for example. This causes the display to look unnatural due to the white point discrepancy. Adaptive Color automatically adjusts a device's display white point to match its environment using an ambient color sensor.



Here's why implementing **Adaptive Color** is beneficial:

- **Consistent Visual Experience:** Adaptive Color uses advanced sensors to adjust your display's color and intensity to match the ambient light. Whether you're working under the harsh fluorescents of an office or the soft glow of a reading lamp, Adaptive Color ensures the content on your screen looks natural and is comfortable to view at all times.
- **Reduced Eye Strain:** By dynamically adjusting to the environment, Adaptive Color can help minimize eye strain, making it ideal for extended use. Whether you're reading, browsing, or working, Adaptive Color makes the display appear more natural, akin to reading a book under different lighting conditions.
- **Enhanced Productivity:** A display that adapts to your environment can lead to less fatigue and more comfortable working conditions, which in turn can increase productivity. Adaptive Color supports this by ensuring that colors and brightness are always optimal for your current setting.
- **A Smarter Display:** Adaptive Color is like having an intelligent assistant for your screen, one that is constantly working to give you the best possible viewing experience. It's a subtle effect that you might not notice until it's turned off, but once you experience it, you'll wonder how you ever did without it.

Implementing Adaptive Color is not just about keeping up with technology; it's about providing a superior user experience that prioritizes comfort and productivity. As you consider the benefits, think of Adaptive Color as an investment in the well-being of your eyes and the quality of your work.

Adaptive Color was originally released in Windows 10, Version 1903. It is available on Windows 10 and Windows 11.

Terminology

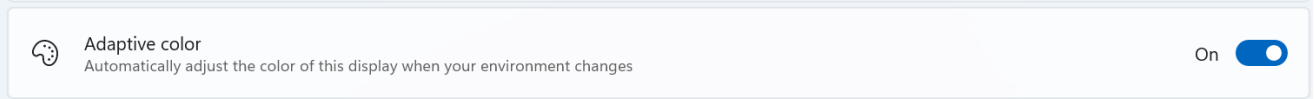
Expand table

Term	Definition
TCON	Timing controller, a subsystem component of a Display panel module.
ACS	Ambient Color Sensor, an ambient light sensor that detects the ambient color levels and relays them to Windows; these sensors must be calibrated and meet certain requirements (see Ambient Light Sensor)
DDI	Device Driver Interface
DES	Display Enhancement Service, an inbox Windows service responsible for managing the display luminance and color adaptation in response to ambient light and ambient color sensor inputs.
CCT	Correlated Color Temperature, a gauge of how yellow or blue the color of light emitted from the display appears. It is measured in Kelvin i.e. 2200K.
CIE 1931 Color space	International Commission on Illumination standardized color space is referenced in this documentation.
ICC	An ICC profile is a set of data that characterizes a color input or output device, or a color space, according to standards promulgated by the International Color Consortium (ICC).
D65	The chromaticity of x=0.31271,y=0.32902

Overview

User Settings

The following setting will be visible to users with Adaptive Color capable systems. For systems that support Adaptive Color, a new option in the **Display Settings Page** under "Brightness and Color" will be exposed. This new option, Adaptive Color, will contain and on/off toggle. There will be brief text inline informing the user of what this new feature is. Adaptive Color is on by default for systems with Ambient Color Sensors (ACS), it uses these sensors to adjust the color and intensity of your display to match the ambient light, so that images appear more natural; adjusting the white point of a display to match the current ambient lighting environment as measured by supported Ambient Color Sensors (ACS).



> **Note**
>
> Adaptive Color currently only supports internal displays. This toggle will only be visible when the internal panel is selected.

System Architecture

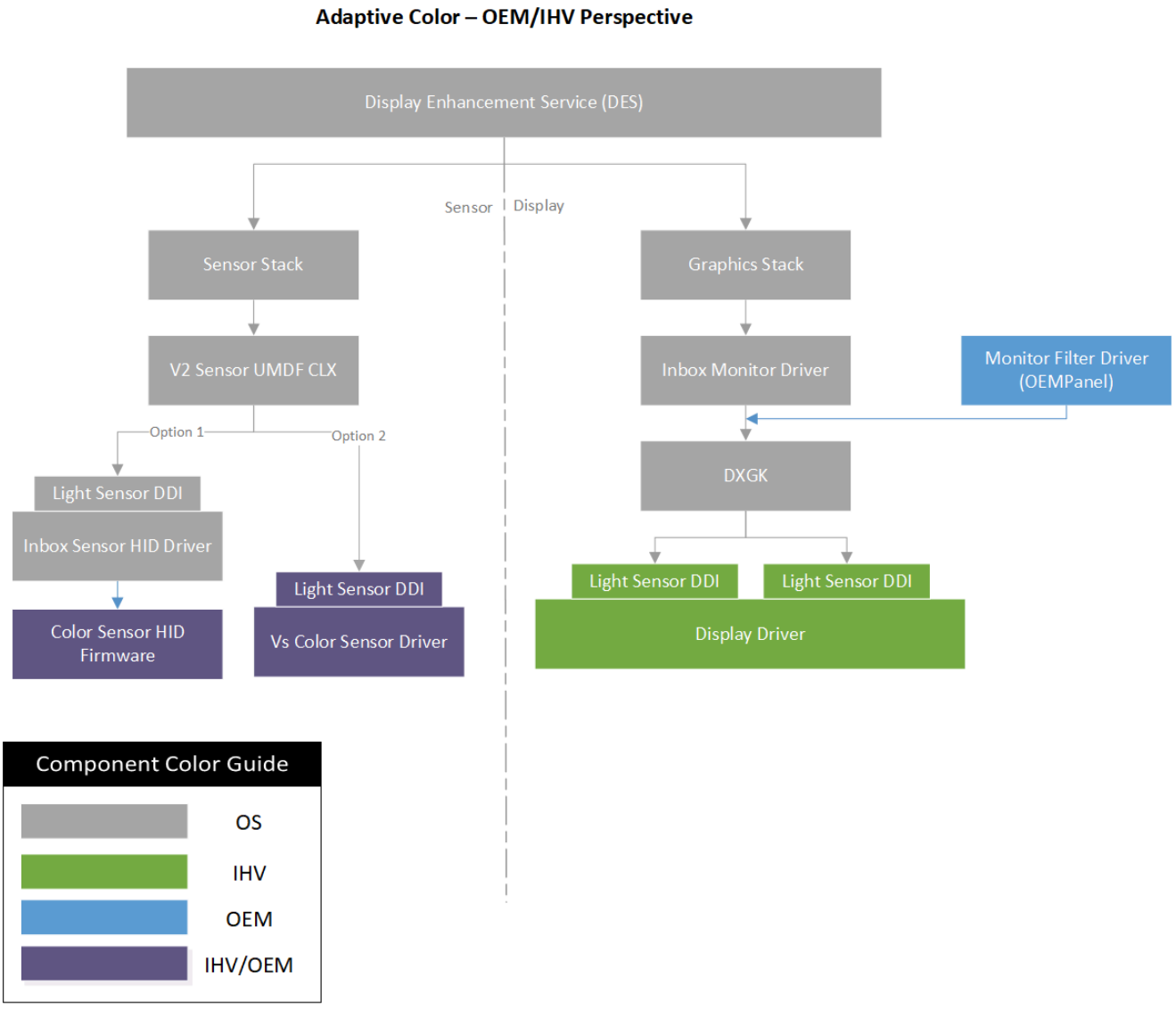
Adaptive Color abides by these principles:

- Adjust the white point of the display to appear more accurate and paper-like in its current environment.
- Slow, non-distracting color transitions when moving between different color environments.
- Hysteresis to prevent excessively frequent white point adjustments.

Illustration below of the improved Adaptive Color implementation and integration with Nightlight shipping in Windows 11, Version 21H2. From top left to bottom right: Adaptive Color adaption for ambient lighting of 9300K [blue light] with a 50% strength slider for Night light, the original image with no nightlight or Adaptive Color adaptation, Adaptive Color adaption for green ambient lighting with a 50% strength slider for Night light, Adaptive Color adaption for Ambient lighting of Illuminant A [Standard Incandescent lighting] with a 50% strength slider for Night light, Adaptive Color adaption for Ambient magenta lighting with a 50% strength slider for Night light.

Ambient: 9300K, NL: 50%
Original image
Ambient: green, NL: 50%
Ambient: illuminant A, NL: 50%
NL becomes less dominant, better preserve the original tint of the ambient environment.
Ambient: Magenta, NL: 50%

The following diagram outlines the various components that work together to enable Adaptive Color on Windows. Ownership is split between the operating system (OS), independent hardware vendor (IHV), and original equipment manufacturer (OEM).



Hardware Requirements

The Adaptive Color feature will be available to the user if the following three requirements are met: a Brightness3 capable (nits-calibrated) display, a supported Ambient Color Sensor (ACS), and a supported OS build.

Display Requirements

The Graphics driver must implement Brightness3 DDI:

- Must support Nits Calibrated Display
- Display must be calibrated

> ⓘ **Note**
>
> Windows features that require Brightness3 require that a driver provide "calibrated" mode support. There are various options for how "calibrated" Brightness3 can be provisioned:
>
> 1. Calibrated display (IHV) driver
> 2. Hybrid calibrated OEM Panel filter driver + calibrated display driver
> 3. Hybrid calibrated OEM Panel filter driver + uncalibrated display driver
> 4. OEM Panel filter driver directly controlling the display

Ambient Light Sensor Requirements

Light sensor DDI is implemented with color support (see Ambient Light Sensor Component Guidelines for details)

- Supports the chromaticity x/y fields and reports the DEVPKEY_LightSensor_ColorCapable property according to the light sensor DDI
- Additionally, if there are multiple color sensors exposed, one sensor must be tagged with the DEVPKEY_LightSensor_AdaptiveColorPreferred property so the OS knows which one to use for Adaptive Color
- HLK requirement System.Client.AmbientLightSensor.Color Calibration requires that the detected chromaticity x and y values are within 0.025 of the actual incoming light. For a full list of requirements see https://learn.microsoft.com/en-us/windows-hardware/test/hlk/testref/system-client-tests.
- Sensor Explorercan be used to validate data from the light sensor.

Interaction with Night Light

Adaptive Color and Night Light are features that work together to adjust the color and brightness of your display to match the ambient light, making images appear more natural. Here's a summary of their relationship and role:

- **Adaptive Color** is a feature that uses Ambient Color Sensors (ACS) to adjust the color and intensity of your display to match the ambient light, so that images appear more natural. It adjusts the white point of a display to match the current ambient lighting environment as measured by supported ACS.
- **Night Light** is a feature that reduces the amount of blue light emitted by the screen, which can help improve sleep quality if used before bedtime.

Adaptive Color fusion with Nightlight

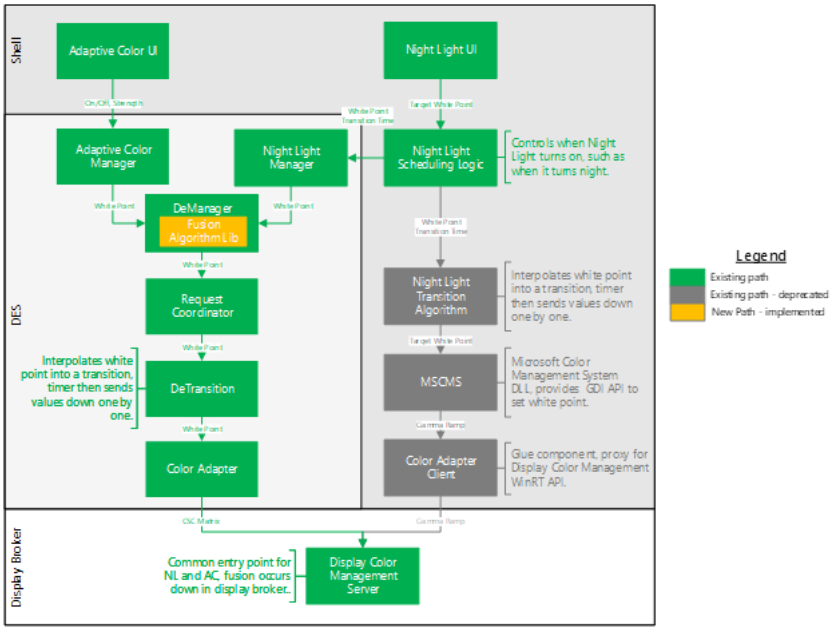
The Night Light/Adaptive Color interaction has been improved in Windows 11. Previously, when both were enabled the screen would turn extremely red as both outputs were multiplied together. The new logic approximates the amount of blue light reduction from the Night Light and Adaptive Color white points and then applies one of two options:

- If the Adaptive Color blue light reduction is stronger - use the Adaptive Color white point as is since it is already achieving the same or stronger blue light reduction as Night Light.
- If the Night Light blue light reduction is stronger - shift the Adaptive Color white point such that its blue light reduction roughly matches Night Light's

Expand table

Windows 10, Version 1903 (9300K+ High Nightlight [2800K])	Windows 11, Version 21H2 (9300K+ High Nightlight [2800K])

Below is the architecture diagram illustrating the changes made in the Display Enhancement Service (DES) for the Windows 11, Version 21H2 to integrate Nightlight and Adaptive Color. Note this only applies to Adaptive Color capable systems. Non-AC devices make use of the 'deprecated' path.


Shell
Adaptive Color UI
Night Light UI
Adaptive Color Manager
Night Light Manager
Night Light Scheduling Logic
Controls when Night Light turns on, such as when it turns night.
DES
DeManager
Fusion Algorithm Lib
Request Coordinator
Interpolates white point into a transition, timer then sends values down one by one.
DeTransition
Color Adapter
Night Light Transition Algorithm
Interpolates white point into a transition, timer then sends values down one by one.
MSCMS
Microsoft Color Management System DLL provides GDI API to set white point.
Color Adapter Client
Glue component, proxy for Display Color Management WinRT API.
Display Broker
Common entry point for NL and AC, fusion occurs down in display broker.
Display Color Management Server
Legend
Existing path
Existing path - deprecated
New Path - implemented

Adaptive Color Algorithm

Transition Time

The color transition algorithm has been improved to be smoother and subtler:

- The transition time will now vary based on the change in white point.
  - Transition time will be shortened if the delta is too small, saving power and ensures the steps appear continuous.
  - Transition time will be increased if the delta is large to ensure users do not notice the change occurring.
- The transition time between two intermittent transition points has been increased to save power with no visual fidelity loss.
- The transition curve will now accelerate and decelerate according to human color perception to ensure the color change remains perceptually consistent.
- The transition curve will now attempt to better minimize hue shift leading to colors looking more accurate while undergoing a transition.

When the feature is turned on (on laptop form factors) there will be a quick, but subtle transition to the adapted white point. Previously there was no transition and the display's white point could appear to "flicker" to the new white point.

White point Clamping

Windows has implemented a "clamping" of the color gamut range of the white point , limiting the range of color adaptation that the user can experience when Adaptive Color is enabled to avoid loss in image fidelity. This means that in extreme lighting conditions, the user's experience will be more pleasant. See below for an example.

Expand table

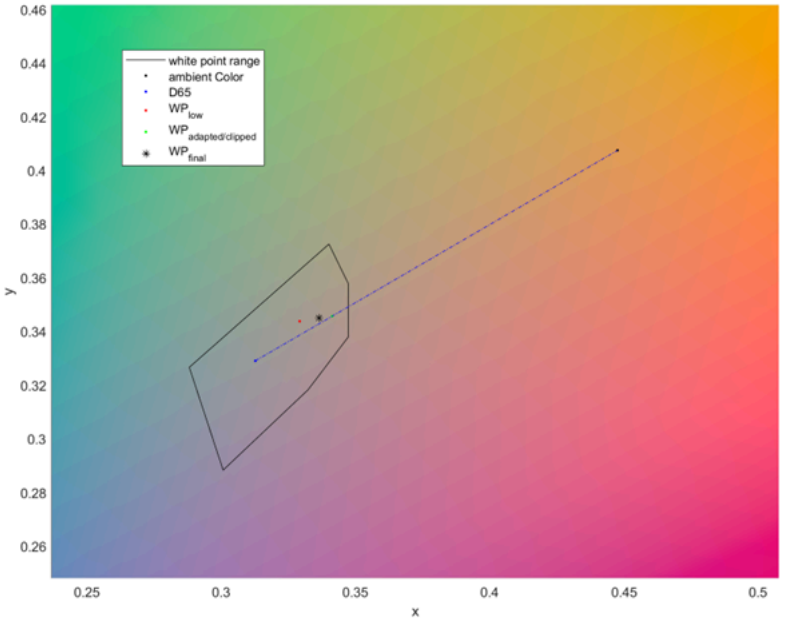
Windows 10, Version 1903 (Extreme Green Environment)	Windows 11, Version 21H2 (Extreme Green Environment)

Adaptive Color follows the steps below to decide the adapted display WP:

1. Determine the low lux WP (WP_{low}), which is a weighted average of D50 and D65 based on the DefaultAdaptiveColorAdaptationStrength (or the slider value if it is presented). Higher strength means the low lux WP will be closer to D50.
2. As descripted in the LuxBasedStrengthAdaptationLut section, the final strength will be the product of the DefaultAdaptiveColorAdaptationStrength/SliderValue and the lux based strength.
3. Based on the final strength, the adapted WP ($WP_{adapted}$) will be a weighted average between the profile default white point (most of the time it is D65) and the ACS CIE xy readings. higher strength means that the adapted WP will be closer to the ACS CIE xy readings.
4. Compare $WP_{adapted}$ with the white point range we defined, if it is beyond the white point gamut, it will be clipped to the gamut boundary to get the clipped WP ($WP_{clipped}$). If $WP_{adapted}$ is within the white point range we defined, $WP_{clipped}$=$WP_{adapted}$.
5. If the ACS lux reading is below the WhitePointMappingLowLightLuxLevel, low light processing will be used to get the final WP. the final WP (WP_{final}) is a weighted average between low lux WP (WP_{low}) defined in (0) and $WP_{clipped}$.

An example is shown in the illustration below when the DefaultAdaptiveColorAdaptationStrength is set to 50% and ACS reading is CIEx= 0.44757, CIE y=0.40745 (illuminant A) and Lux=30.

1. First WP_{low} (x=0.3292, y=0.3438) is determined, which is 50% D65 and 50% D50.
2. Then the final strength is determined as $S=S_L*S_D=25\%$.
3. $WP_{adapted}$ will be the weighted average of D65 and ambient color ACS value.
4. Compare $WP_{adapted}$(x=0.3413,y=0.3457) with the white point range, it is within the range, so no clipping will be applied and $WP_{clipped}$=$WP_{adapted}$.
5. As the ACS lux (30) is below the WhitePointMappingLowLightLuxLevel (50), low lux processing will be applied. WP_{final} (x=0.3365,y=0.3449) is 30/50*100%=60% $WP_{clipped}$ and 40% WP_{low}.



Sample Output

Adaptive Color Algorithm Output Table (**using the default settings**) can be found below:

The sample output of the Adaptive Color algorithm in DES for a set of 10 ambient lighting conditions

Expand table

ACS Reading: CIE x	ACS Reading: CIE y	ACS Reading: Lux	CCT	Display CIE x	Display CIE y	Display CCT
0.4593	0.4107	5	2706	0.3306	0.3440	5580
0.4593	0.4107	5	2706	0.3306	0.3440	5580
0.4593	0.4107	50	2706	0.3436	0.3462	5045
0.3805	0.3767	30	4008	0.3285	0.3414	5677
0.3805	0.3767	100	4008	0.3443	0.3512	5032
0.3457	0.3585	50	5002	0.3204	0.3359	6067
0.3457	0.3585	500	5002	0.3285	0.3431	5675
0.3127	0.3290	100	6504	0.3127	0.3290	6504
0.3127	0.3290	1000	6504	0.3127	0.3290	6504
0.3038	0.3140	500	7194	0.3082	0.3213	6831
0.3038	0.3140	1200	7194	0.3127	0.3290	6504
0.2849	0.2932	700	9224	0.2983	0.3105	7632
0.2849	0.2932	1200	9224	0.3127	0.3290	6504

OEM Customizations

The following table outlines customizations that can be made by OEMs. It also lists the default value which is used if not customized.

The registry path for all public OEM settings is located under this registry path: `HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\AdaptiveDisplayBrightness\{23B44AF2-78CE-4943-81DF-89817E8D23FD}`

Expand table

Key	Description	Default	Fo
AllowAdaptiveColorCapable	If the device can expose Adaptive Color capability.	1	R
EnableAdaptiveColorStrengthSlider	The slider for strength will not show up. We removed the slider starting in the Windows 11, Version 21H2, to improve the consistency of the experience. For example if a user sets the strength for a particular lighting environment they may experience poor adaption when moving to a different environment (ex. Low-lighting). Systems without the slider will default to 50% strength. Systems upgraded to the Window October 2021 release, will be the strength reset to 50% and the slider will be removed.	FALSE	R
DefaultAdaptiveColorAdaptationStrength	How strong the adaptation strength is for Adaptive Color by default. Will be ignored when slider is disabled.	50	R

Key	Description	Default	F
WhitePointMappingLowLightLuxLevel	The low lux level below which Adaptive Color will not adapt according to the ACS since color sensors are typically inaccurate at these levels. Adaptive Color will instead output a weighted average between the static D50 (x=0.34567,y=0.35850) and the ACS value, lower lux level means the Adaptive Color will put more weight on D50.	5000	R
LuxBasedStrengthAdaptationLut	Look-up table used to map a particular millilux value to a strength multiplier that gets combined with the current strength setting and then used for color adaptation. More details in LuxBasedStrengthAdaptationLut section above.	`0:0.5,50000:0.5,60000:0.6,70000:0.7:80000:0.8,90000:0.9,100000:1.0,600000:1.0, 650000: 0.9958,`	R
ColorTransitionMinTransitionTime	The minimum allowed transition time, used by the algorithm to calculate the number of transition points.	300	R
ColorTransitionBaseTransitionTime	The default interval used for color transitions to a specific whitepoint.	400	R
IsAdaptiveColorOnByDefault	If Adaptive Color is turned on or off by default	True	R

OEM Customization - LuxBasedStrengthAdaptationLut

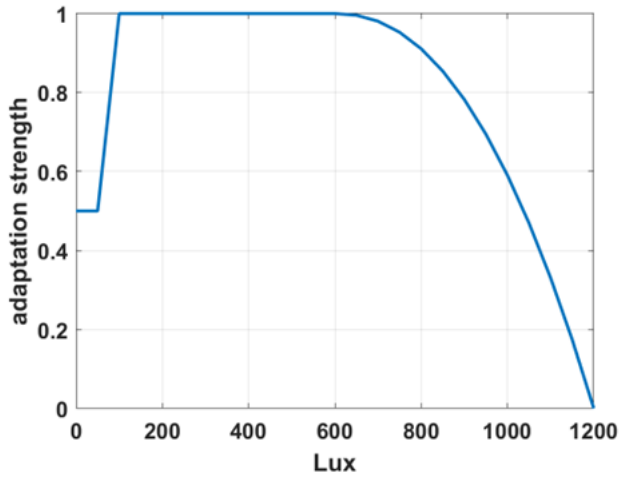
In Adaptive Color, adaptation strength is used to determine how much we would like the display white point (WP) to follow the ambient color. 100% adaptation strength means the display WP will be the same as the ambient color sensor reading (in terms of CIE x,y) and 0% means no adaptation strength will be applied and the display will always be at the default white point whatever the ambient lighting conditions.

For a real world ambient color sensor (ACS), they usually have different accuracy under different Lux levels. For example, at low lux level the accuracy of ACS is usually low because of the low signal to noise ratio. At the same time, at high lux levels human perception is dominated by ambient light and changing the display color do not have much impact on the human perception. Because of these reasons, the LuxBasedStrengthAdaptationLut is introduced to control the adaptation strength at different Lux levels. As it shown below, at 0 Lux the lux based strength (S_L) is 0.5 (50%) and at 300 lux S_L is 1 (100%).

If the slider is not presented, the final strength (S) applied is the product of S_L and DefaultAdaptiveColorAdaptationStrength (S_D). For example, if S_D is set to 50%, based on the Lut below at 0 Lux the final strength is 25% and at 300 Lux the final strength is 50%.

If the user controllable slider is presented, the final strength is the product of S_L and slider value (in percentage). For example, if the slider is set to 70%, based on the Lut below at 0 Lux the final strength is 35% and at 300 Lux the final strength is 70%.

OEMs can customize this luxBasedStrengthAdaptationLut based on the ACS performance (see below).



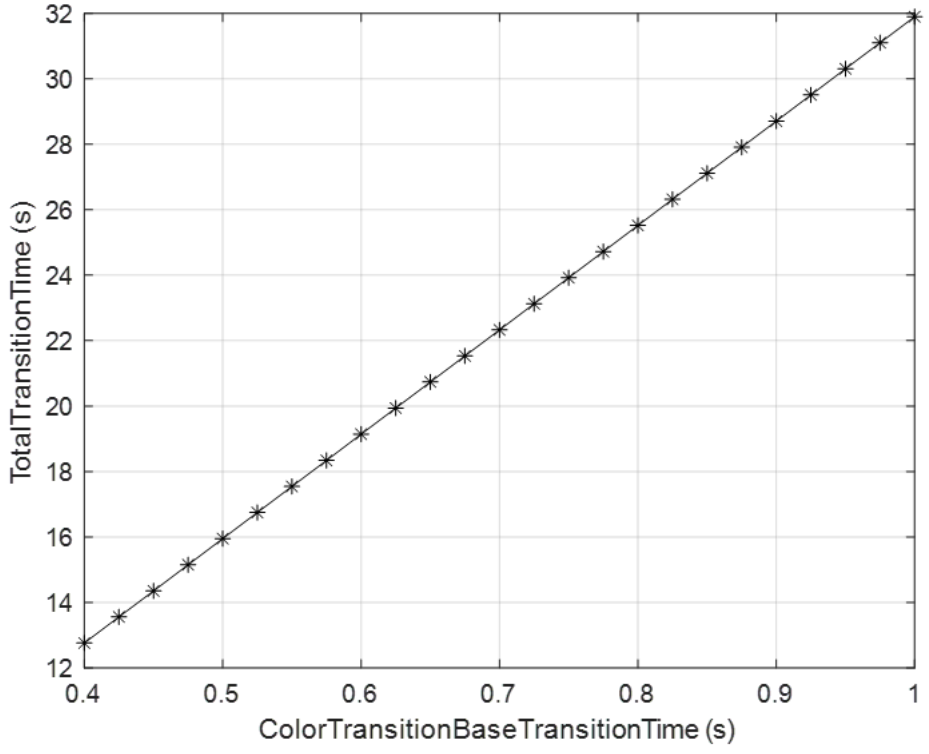
OEM Customization - ColorTransitionMinTransitionTime

When the ambient color changes, DES will kick off a transition from current display WP to the new display WP and send each transition point to lower graphics stack one by one. However, if the transition points are sent out too frequently (for example, every 10 ms), the hardware will not be able to consume each transition point and apply the color changes accordingly. The ColorTransitionMinTransitionTime is the minimum time interval between two adjacent transtion points.

OEMs should customize the ColorTransitionMinTransitionTime based on hardware capabilities.

OEM Customization - ColorTransitionBaseTransitionTime

The totalTransitionTime is proportional to the ColorTransitionBaseTransitionTime. Below is an example at 300 Lux where the ambient color changes from D65 to D50. The total transition time increases from 12.76s to 32s when ColorTransitionBaseTransitionTime increases from 400ms to 1s.



OEMs can customize the ColorTransitionBaseTransitionTime to tune how long the transition takes.

Override Options

In the case that an app does not want the color adjustments Adaptive Color provides we have extended the Display Enhancement Override (DEO) to include DisplayColorOverrideScenario for Adaptive Color.

Validation Test Procedure

The purpose of this document is to define the manual validation list for Adaptive Color. For reference, this document will refer to the device under test as "DUT". See Sensor Testing for more tests.

OEM Settings

Configure the DUT with the Adaptive Color OEM settings the device will be shipped with. These settings are listed in the table below for reference, for a more detailed table refer to Table 1:

Expand table

OEM Setting	Default Behavior	Expected Changed Behavior
Default Adaptive Color On/Off State	Adaptive Color is turned off by default.	When the value *1* is specified, Adaptive Color should be turned on by default. When the value *0* is specified, Adaptive Color should be turned off by default.
Default Adaptive Color Adaptation Strength	Adaptive Color adaptation strength slider position should be at 50% in the display settings page by default.	When a number *X* in range [0, 100] has been specified, the Adaptive Color slider position should be at *X*% in the display settings page by default.

Run the following validation steps:

1. Ensure the default behavior is working as expected when the OEM setting is not specified:
   a. If present, remove the OEM setting registry key on the DUT.
   b. Reboot the DUT.
   c. Validate the default behavior, as listed in the table above, for the OEM setting has taken effect.

2. Ensure the OEM setting is being applied properly when it has been specified:
   a. Add the OEM setting registry key on the DUT.
   b. Reboot the DUT.
   c. Validate the expected OEM setting behavior, as listed in the table above, has taken effect.

OEM settings only take effect when the user has not explicitly modified the value yet. For example, if the user has manually turned on Adaptive Color, then the OEM setting for *Default Adaptive Color On/Off State* will be ignored. So, if the user has touched the Adaptive Color settings, the DUT will need to be clean flashed to properly test the OEM setting scenarios.

Upgrade Scenarios

1. Ensure devices that meet the Adaptive Color hardware requirements gain the Adaptive Color feature when upgraded from RS5 (which does not support Adaptive Color) to 19H1 (which does support Adaptive Color):
   a. Clean install the latest RS5 release build (or OEM image, if applicable) on the DUT.
   b. Upgrade to 19H1 and install any applicable driver updates[1].
   c. Validate the Adaptive Color feature shows up in the display settings page and follows the Adaptive Color behavior defined in this document.

2. Ensure devices that support Adaptive Color with a 19H1 release build still support Adaptive Color after upgrading to any later 19H1 release build.

3. Ensure OEM settings are migrated properly when upgraded:
   a. Add the OEM settings as specified in the "OEM Settings" section.
   b. Upgrade the DUT to a newer 19H1 release build.
   c. Verify the OEM settings are still present and taking effect as expected.

4. Ensure Adaptive Color settings are persisted
   a. Turn Adaptive Color on.
   b. Upgrade the DUT to a newer 19H1 release build.
   c. Verify Adaptive Color is still on and taking effect as expected.
   d. Repeat steps a) to c) for Adaptive Color off and new Adaptive Color strength values.

The Adaptive Color requirements are (1) expose a color sensor and (2) expose the Brightness3 DDI.

Adaptive Color UI

1. Ensure Adaptive Color takes effect when turned on from display settings page:
   a. Ensure Adaptive Color is turned off.
   b. Go into a non-D65 (https://en.wikipedia.org/wiki/Illuminant_D65)lighting environment (normal office lighting works fine).
   c. Turn on Adaptive Color in the display settings page.
   d. Validate the display gets warmer https://en.wikipedia.org/wiki/Color_temperature .
   e. Turn off Adaptive Color in the display settings page.
   f. Validate the display returns to the same color temperature in step a.

2. Ensure Adaptive Color respects the adaptation strength slider (IF enabled in OEM Customizations, see OEM Customizations):
   a. Go into a non-D65 (https://en.wikipedia.org/wiki/Illuminant_D65) lighting environment (normal office lighting works fine).
   b. Turn on Adaptive Color in the display settings page.
   c. Validate the display gets whiter when moving the Adaptive Color adaptation strength slider closer to 0%.
   d. Validate the display gets warmer when moving the Adaptive Color adaptation strength slider closer to 100%.
   e. Turn off Adaptive Color in the display settings page.
   f. Validate the display returns to the same color temperature in step a.

Adaptive Color Capability Check

Ensure the Adaptive Color feature shows up in the Display Settings UI when the device exposes a brightness3 and ambient color sensors interface and the AllowAdaptiveColorCapable OEM customization is enabled.

Persistent Adaptive Color Settings

1. Ensure user preferences for Adaptive Color on/off are being persisted:
   a. Turn off Adaptive Color in the display settings page.
   b. Reboot the DUT.
   c. Validate the Adaptive Color toggle in the display settings page is still turned off.
   d. Repeat steps 1 to 3, but with Adaptive Color turned on.
   e. Repeat steps 1 to 4, but with hibernate and suspend instead of rebooting in step b.

2. Ensure user preferences are stored for Adaptive Color adaptation strength :
   a. Select a random number *X* in range [0, 100] and move the Adaptive Color adaptation strength slider to *X*% in the display settings page.
   b. Reboot the DUT.
   c. Validate the Adaptive Color adaptation strength slider in the display settings page is still at *X*%.
   d. Repeat steps a to c several times with new slider values.
   e. Repeat steps a to d, but with hibernate and suspend instead of rebooting in step b.

Night Light Interaction

1. Ensure Night Light and Adaptive Color don't affect each other negatively:
   a. Turn on Adaptive Color in the display settings page.
   b. Turn on Night Light in the display settings page and ensure it is active:
   c. Validate the display gets warmer after Night Light is turned on and the display remains in a usable state.
   d. Repeat steps a. to c. with different Night Light slider strengths and Adaptive Color adaptation slider strengths.

Color Lighting Scenarios

1. Ensure Adaptive Color looks pleasing in different common lighting scenarios (see Table 2 Adaptive Color Output - The sample output of the Adaptive Color algorithm in DES for a set of 10 ambient lighting conditions):

   a. Turn on Adaptive Color in the display settings page.

   b. Move the Adaptive Color adaptation strength slider to 50%.

   c. Use the device in various color lighting conditions and validate that the display is pleasing to look at. Common lighting scenarios (https://en.wikipedia.org/wiki/Color_temperature) (from warmest to whitest):
      i. Candle/firepit (around 1800 Kelvins)
      ii. Sunset/sunrise (around 1800 Kelvins)
      iii. Incandescent lamp (around 2500 Kelvins)
      iv. Warm LED lamp (around 3000 Kelvins)
      v. Fluorescent lamp (around 5000 Kelvins)

vi. CFL lamp (around 5000 Kelvins)
vii. Daylight (around 6500 Kelvins)
viii. Overcast daylight (around 6500 Kelvins)

2. Ensure Adaptive Color does not behave eccentrically under abnormal lighting scenarios:
   a. Turn on Adaptive Color in the display settings page.
   b. Move the Adaptive Color adaptation strength slider to 50%.
   c. Move into an abnormal lighting scenario. This is any lighting condition that is not encountered day-to-day. For example, a magenta color LED lamp.
   d. Validate that the display doesn't shift to the eccentric color.

3. Ensure Adaptive Color does not flicker under strobe like lighting conditions:
   a. Go into a non-D65 lighting environment with a controllable on/off light source (normal office lighting works fine).
   b. Turn on Adaptive Color in the display settings page.
   c. Move the Adaptive Color adaptation strength slider to 50%.
   d. Quickly turn the light sources on/off repeatedly with an interval of one second or less.
   e. Validate that the display colors do not flicker with the light, it should stay consistent throughout the scenario.

4. Ensure Adaptive Color has pleasing color transitions:
   a. Go into a lighting environment with two light sources of different color temperatures that can be turned on/off. For example, office lights and an incandescent lamp.
   b. Turn on Adaptive Color in the display settings page.
   c. Move the Adaptive Color adaptation strength slider to 50%.
   d. Turn the first light source on and the second light source off.
   e. Wait for Adaptive Color to stabilize to the first light source.
   f. Turn off the first light source and turn on the second light source.
   g. Validate Adaptive Color adapts to the new lighting source with a smooth and slow color transition.

5. Ensure Adaptive Color adapts to new lighting scenarios in lid open scenarios for clamshell devices:
   a. Ensure Adaptive Color is turned on.
   b. Close the lid of the device.
   c. Go into a new lighting environment that has a different whitepoint.
   d. Open the lid of the device and verify the whitepoint has quickly (< 1 second) changed to the new lighting environment.

Auto Brightness Interaction

1. Ensure auto brightness and Adaptive Color don't affect each other negatively:
   a. Turn Adaptive Color and auto brightness on in the display settings page.
   b. Use the device in various brightness and color lighting conditions, the display should be pleasing to look at in all scenarios. The common scenarios here are the same as listed in the *Color Lighting Scenarios* section.

Color Filter Interaction

1. Ensure the color filter conflict messages are shown at expected times:
   a. Turn on a color filter in the color filters display page:
   b. Go to the Adaptive Color display settings page and ensure the warning text about color filter conflicts appears:
   c. Go back to the color filter page and turn off any color filters
   d. Go to the Adaptive Color display settings page and ensure the warning text is now yellow:
   e. Plug in an external monitor to the DUT and validate steps a to d again.

2. Ensure Adaptive Color and color filters work nicely together visually:
   a. Go into a non-D65 lighting environment (normal office lighting works fine).
   b. Turn on one of the color filters:
   c. Turn on Adaptive Color in the display settings page.
   d. Move the Adaptive Color adaptation strength slider to 50%.
   e. Validate the display still looks like the color filter is applied but is warmer.
   f. Repeat steps a to e with each of the color filter options.

In 19H1, color filters used in combination with Adaptive Color is not guaranteed to help color blindness. This validation scenario is simply ensuring the display doesn't go into a wildly unexpected state when Adaptive Color and color filters are both turned on.

Display Enhancement Override WinRT API Interaction

1. Ensure Adaptive Color can be temporarily overridden with the Display Enhancement Override (DEO) WinRT API
   a. Download the SensorExplorer app (www.aka.ms/sensorexplorer) and install it.
   b. Go into a non-D65 lighting environment (normal office lighting works fine).
   c. Turn on Adaptive Color in the display settings page.
   d. Move the Adaptive Color adaptation strength slider to 50%.
   e. Engage the accurate color override via SensorExplorer.
   f. Verify Adaptive Color is no longer taking effect, the display should look as if Adaptive Color is off.
   g. Stop the color override from SensorExplorer.
   h. Verify Adaptive Color has returned with the same warmness as before in step #4.

2. Sanity check apps which consume DEO work as expected
   a. Obtain a device that supports HDR and Adaptive Color.
   b. Go into a non-D65 lighting environment (normal office lighting works fine).
   c. Turn on Adaptive Color in the display settings page and verify the display gets warmer.
   d. Open the Movies & TV app and play any video (a movie trailer works fine).
   e. Verify Adaptive Color isn't taking effect during video playback.
   f. Close the Movies & TV app and verify Adaptive Color takes effect again

Interaction of Adaptive Color with other Windows Display features

Adaptive Color Transitions with Lid Open/Close

On Lid Open event, DES will try to restore the color adaptation target based on the time between the lid-close and lid-open.

- Less than 15 seconds
  - The last known whitepoint target will be restored.
- Greater than 15 seconds and less than 120 seconds
  - The last known white point will be interpolated to based on D65 curve and applied.
- Greater than 120 seconds
  - The initial whitepoint will fall back to D65

Adaptive Color Support on External Monitors

No planned support for Adaptive Color feature for external monitors. Night Light for external monitors will not be affected by the Adaptive Color feature.

Adaptive Color Interaction with HDR Monitors

This feature has not been validated with HDR panels. The feature changes the whitepoint of the monitor regardless of the mode it is operating in.

Adaptive Color interaction with Adaptive Brightness

The Adaptive Color and Adaptive Brightness controls are orthogonal.

Presence sensing

Article • 12/05/2023

With the release of Windows 11, Microsoft now natively supports the presence sensing feature set in Windows. For Windows 11 PCs that have a presence sensor built in, users can have their screen turn off automatically when they leave (see Lock on leave), and then have their device wake up quickly when they approach (see Wake on approach). This can help keep their PC more secure, help save battery power, and help users get back to work more quickly.

> **Tip**
>
> To determine if your device has Microsoft's presence sensing support, check if you have settings as described **in this Support article**. For more information on the data collected by the feature see the **Microsoft Privacy Statement**.

Presence sensing is a set of features introduced in Windows 11, which includes Lock on Leave, Wake on Approach, and Adaptive Dimming . Presence sensors are a new class of proximity sensors that detect humans. This section primarily deals with the end user device experiences and recommendations. Presence sensors can be based on a variety of technology, including computer vision or radar; however, there are requirements to ensure implemetations process data locally and no image metadata is transmitted to the sensor stack. The Windows sensors stack does not leverage any AI models internally in the OS; some third-party presence sensors may make use of AI models. For more details on the hardware requirements see Presence sensors in the hardware component guidelines section. Microsoft's use of AI models is dictated by the Responsible AI standard.

> **Note**
>
> Some OEM devices implement custom sensors which offer similar functionality to presence sensing, however these implementations are not supported by Microsoft and do not benefit from Microsoft inbox features like user app privacy settings.

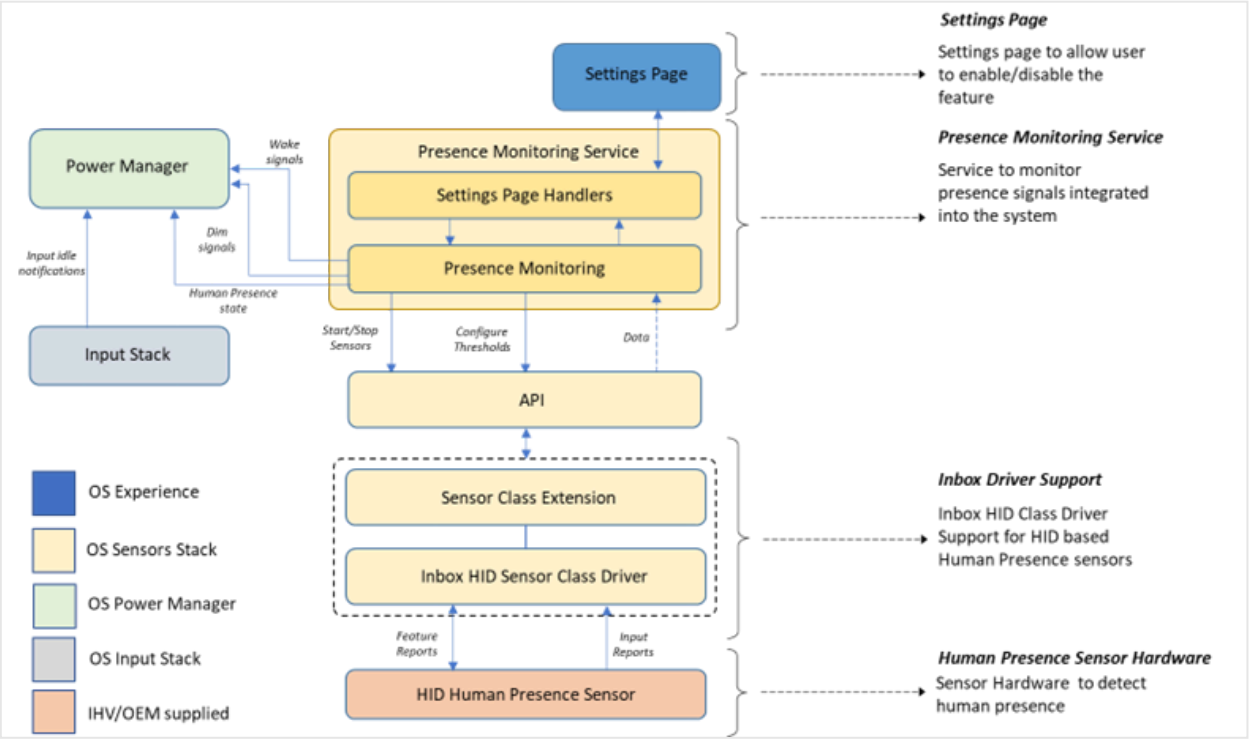
Terminology

Expand table

Term	Definition
Human presence sensor	Any hardware (or aggregation of hardware signals) that can detect one or more human's distance from a device or intent to interact with the device. This was unofficial name for the group of features, the official name is Presence Sensing
Presence sensing	The Windows term to describe the grouping of features of wake on approach, lock on leave, and so on.
Category I: presence sensor	Hardware (or an aggregation of hardware signals) that implements facial presence detection; scanning to detect a face and providing a bounding box around the face. Facial presence detection does not involve distinguishing one face from another face, predicting or classifying facial attributes.
Category II: human presence sensor	Hardware (or an aggregation of hardware signals) that implements people tracking; detecting and tracking individual movements that identify the presence of one or more humans
Distance agnostic	A human presence sensor or camera that does not expose distance to the OS. The hardware only exposes a binary human presence signal.
Proximity sensor	A piece of hardware, exposed separately from the device's camera in firmware, that can detect the user or object's distance from the device and exposes it to the OS. Presence sensors are a subset of human proximity sensors.
Distance-agnostic sensor	A piece of hardware, separate from the device's camera, that can detect a user's inexact location in relation to a device and exposes to the OS only a binary signal of human presence or lack of presence.
Host	Device running Windows
Internal sensor	Hardware installed physically in the host.
External sensor	Hardware physically outside of the host.

Overview (hardware, software, and API)

To enable the human presence features on Windows 11, additional hardware that supports the Microsoft defined presence sensor definition is required. The wake on approach and lock on leave features are available on all releases of Windows 11, while other features will be made available in future versions which will be called out in the document. HID components are optional but recommended and may be replaced with a custom sensors class extension driver.



OEM Customizations

OEMs can configure their devices default behaviours for Presence Sensing with the settings documented in the Unattend reference, see Microsoft-Windows-Sensors-Core for details.

Power Settings

Additionally, there are new Adaptive Power subgroup power settings related to Presence Sensing. See Presence sensing power settings overview

Related topics

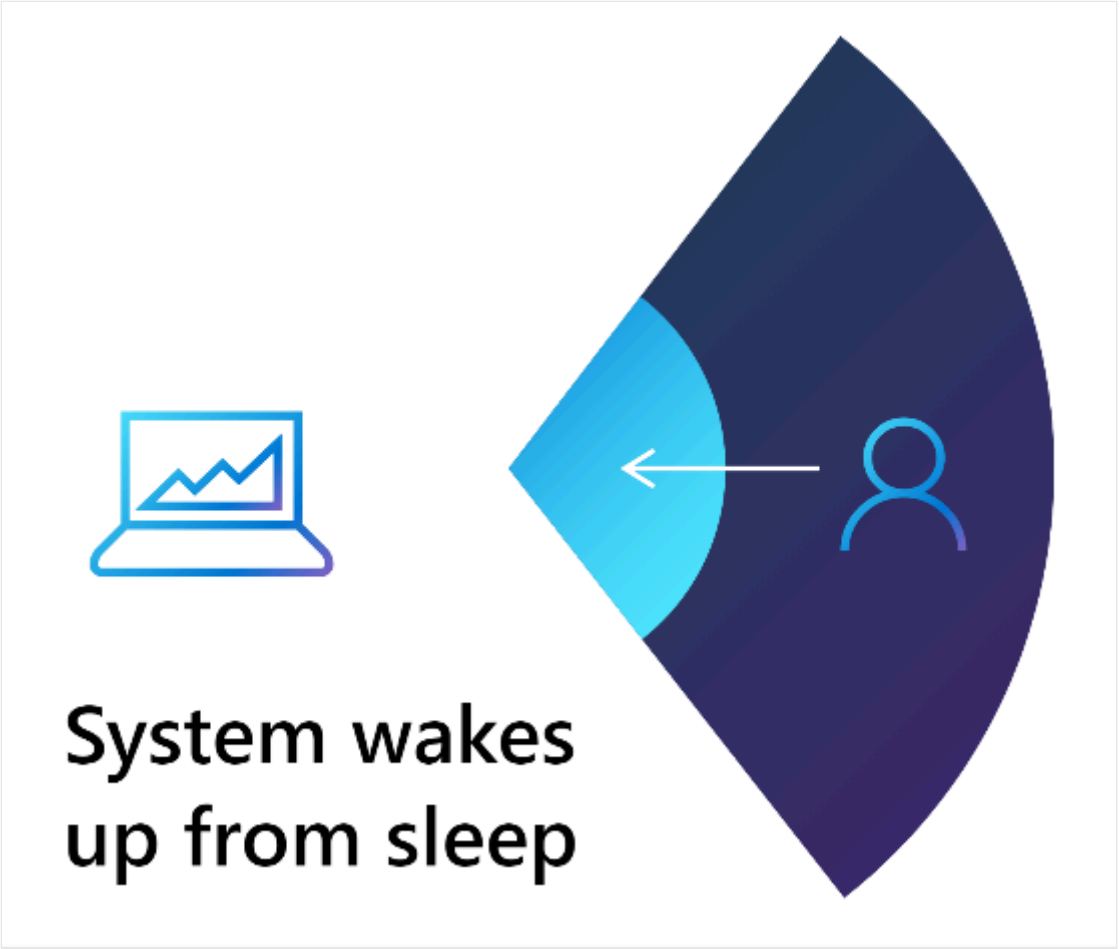
- Presence sensor hardware component guideline
- Presence sensor API documentation
- Sensors driver design guide
- Sensors HID Usages
- Windows Hardware Compatibility Program

Wake on approach

Article • 05/12/2023

This section provides an overview of the various experiences enabled by Windows 11's presence sensing features. Each experience can be enabled independently, however Microsoft encourages the implementation of as many of these features as possible for a given hardware configuration.

Functional overview



Wake on approach: When the user approaches the device, it should recognize the user's intent to initiate use of the device. By the time the user is seated and ready to use the device, it is awake and beginning the authentication process.

Debounce mechanisms

The system has been manually locked, but the user is still present

The state machine tracks if a display on or off event has occurred while a human is detected. If the display has been turned off, which corresponds to lock, then the feature will not wake the system until the display is manually turned back on or user walks away then back. This prevents the system from being immediately woken after manually being put to sleep, such as the user manually locking via the **Windows+L** keyboard shortcut.

User is repeatedly moving in and out of the wake or lock distance threshold

The OS handles this case, no requirements to arbitrate this in firmware. The state machine will wait at least 15 seconds (5 seconds if the timeout is set to 10 seconds) between human presence-based locks to prevent the system from repeatedly switching between wake and lock states. This does not mean that the device will not be able to wake up immediately following a lock. An example of where this can occur is if the user is rocking back and forth in their chair so they repeatedly and quickly enter/exit the wake/lock detection distance. A single repeated motion switching between wake and lock state will trigger the mechanism.

> ⓘ **Note**
>
> It is known that Windows Hello face ID is often combined with wake on approach for improved user experience. When testing the features together, it is known that sometimes Windows Hello face sign-in has a debounce timer of 2 minutes that requires user to dismiss login screen.

Exceptions

Prior to Windows 11 22H2 May 2023 update, the device will only wake from modern standby, not from S3 for HID-based sensors. In Windows 11 22H2 May 2023 update (build 22624 or later), HID-based sensors are able to support waking from lower power states like S3. In Windows 11 24H2 if in a state of excessive power drain, Wake on Approach wake may be disabled while on DC power to preserve battery.

Holistic experience benchmarks

This section defines what the user should experience when using this feature, regardless of the hardware facilitating the experience.

Expand table

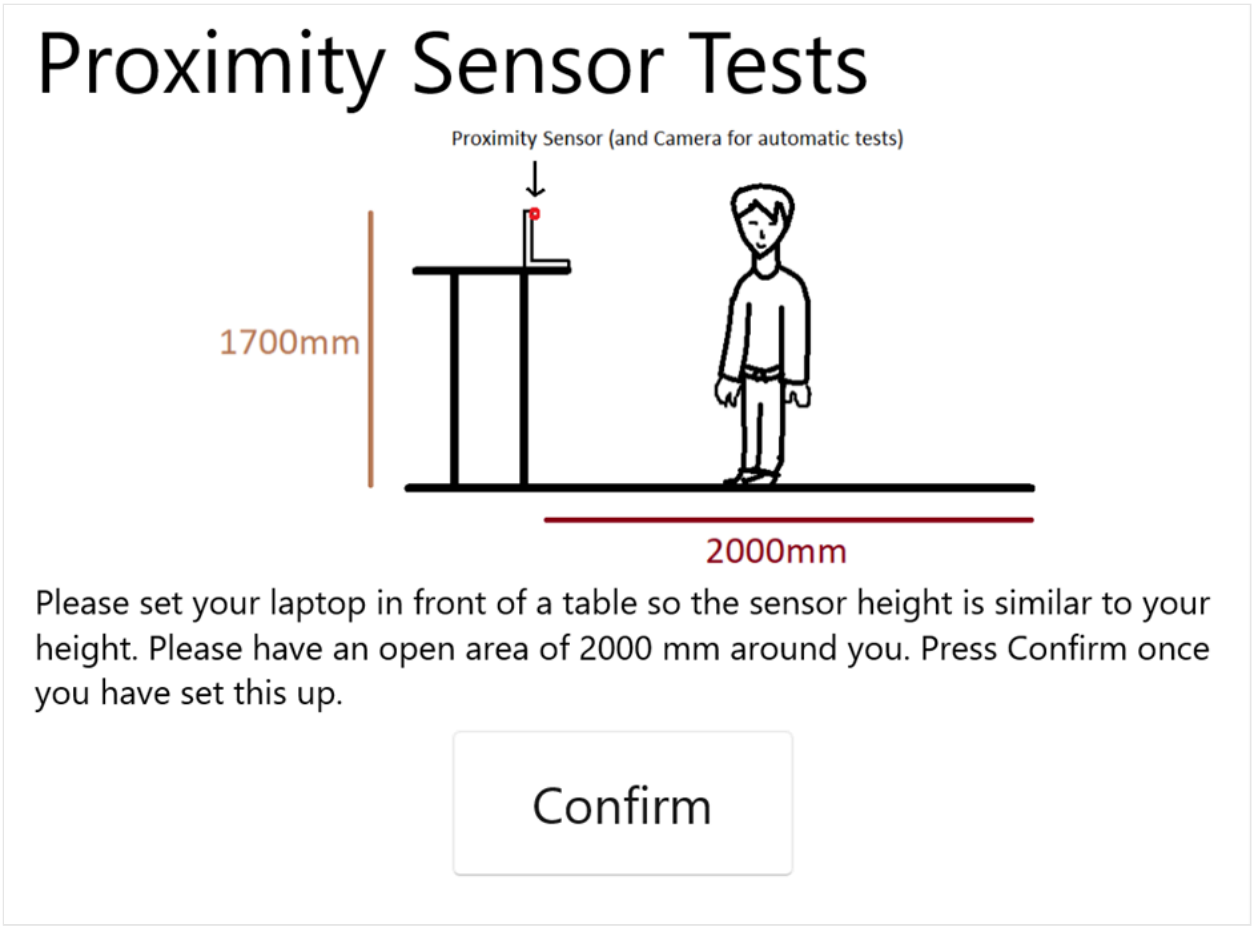
Measure	Metric	Comments
Wake latency – average	<1.0 second average	Defined as the time between when the user reaches the distance directly measured (defined below) at 45 cm (18 in), 75 cm (29.5 in), and 120 cm (47.2 in) and the time when the device's screen turns on. Average of the three trials.
External monitor metrics	Same as internal	This is subject to change based on experimentation and further user research studies

Accuracy – User Intent Recognition Framework:

Expand table

Measure	Description	Target
False Positives	The device wakes, but the user is not actually in front of the device.	<5%
False Negatives	The user is in front of the device, but the device does not wake up.	<5%

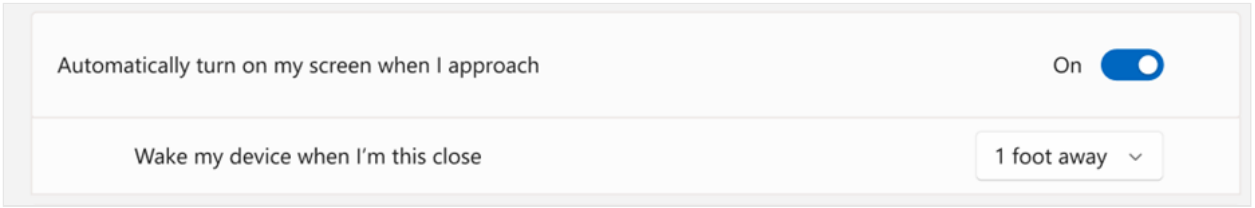
Directly measured: The PC placed on a standard desk, 30 inches (76.2 cm) off the floor. The user approaches the desk and chair from >1.2m away. See Figure below. This concept is only used for consistency of measurement.


Proximity Sensor Tests
Proximity Sensor (and Camera for automatic tests)
1700mm
2000mm
Please set your laptop in front of a table so the sensor height is similar to your height. Please have an open area of 2000 mm around you. Press Confirm once you have set this up.
Confirm

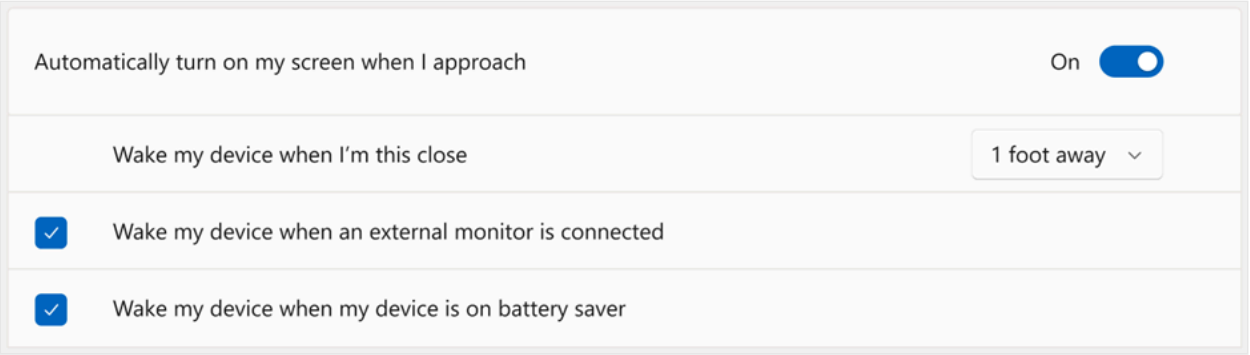
Example settings

The distance setting for Wake on Approach, the distance drop-down, is only available for distance capable devices.

Windows 11, version 21H2



Windows 11, version 22H2 and above


Automatically turn on my screen when I approach
On
Wake my device when I'm this close
1 foot away
Wake my device when an external monitor is connected
Wake my device when my device is on battery saver

Related topics

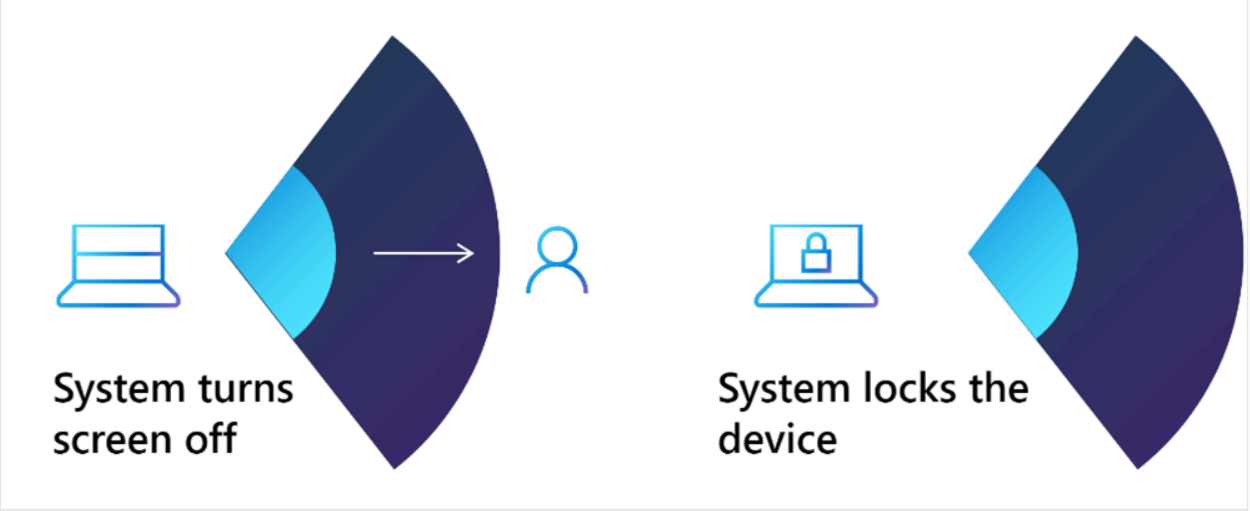
- Presence sensing device experiences
- Presence sensor hardware component guideline
- Presence sensor API documentation
- Sensors driver design guide
- Windows Hardware Compatibility Program

Lock on leave

Article • 12/06/2024

This section provides an overview of the various experiences enabled by Windows 11's presence sensing features. Each experience can be enabled independently, however Microsoft encourages the implementation of as many of these features as possible for a given hardware configuration.

Functional overview



Lock on leave: When the user is no longer actively present in front of the device, the operating system will lock the screen.

> **Note**
>
> The feature does not directly lock the device. The screen is turned off which locks the device after some time. The screen turns off 5 seconds before the selected timeout in HPD Settings page. However, the system locks only after the selected timeout. For example, if the selected timeout is 10 seconds, the system would lock after 10 seconds and an additional preset time.

Debounce mechanisms

The system is manually locked, but the user is still present

The state machine tracks if a display on or off event has occurred while a human is detected. If the display is turned off, which corresponds to lock, then the feature will not

wake the system until the display is manually turned back on or user walks away then back. This prevents the system from being immediately woken after manually being put to sleep, such as the user manually locking via the **Windows+L** keyboard shortcut.

User is repeatedly moving in and out of the wake or lock distance threshold

The OS handles this case, no requirements to arbitrate this in firmware. The state machine will wait at least 15 seconds (5 seconds if the timeout is set to 10 seconds) between human presence-based locks to prevent the system from repeatedly switching between wake and lock states. This does not mean that the device will not be able to wake up immediately following a lock. An example of where this can occur is if the user is rocking back and forth in their chair so they repeatedly and quickly enter/exit the wake/lock detection distance. A single repeated motion switching between wake and lock state will trigger the mechanism.

User wakes device with keyboard or mouse while out of view of presence sensor

In the off-screen state, the person wakes up the screen with the mouse or any key on the keyboard from the side (the person is not within the recognition range), then the screen will not be locked after the person leaves, if they do not re-enter human presence. The human presence sensor needs to see a change from *Present* to *Not Present*.

Exceptions

At certain times, the user does not want their device to lock, even if they are not physically in proximity to the device. In these cases, the operating system should not lock the screen even if there is not a signal of human presence from the human presence sensor.

In any scenario that takes in a power request from the OS, the lock will not be executed. This applies to scenarios such as a video, ongoing call, etc. This behavior will mimic the current power management schema for current timeout-based locks. This is not configurable to the user since these exceptions are only activated at the kernel level.

Conflict with power settings timeout

For the screen timeout options under **System** > **Power and battery**, when human presence is enabled, the more aggressive setting wins. For example, if timeout is set to 1 minute and lock on leave is set to 2 minutes then the screen will turn off in 1 minute whether user is present or not. If timeout is set to never and lock on leave is set to 2 minutes then the screen will turn off after 2 minutes.

Holistic experience benchmarks

This section defines what the user should experience when using this feature, regardless of the hardware facilitating the experience.

Expand table

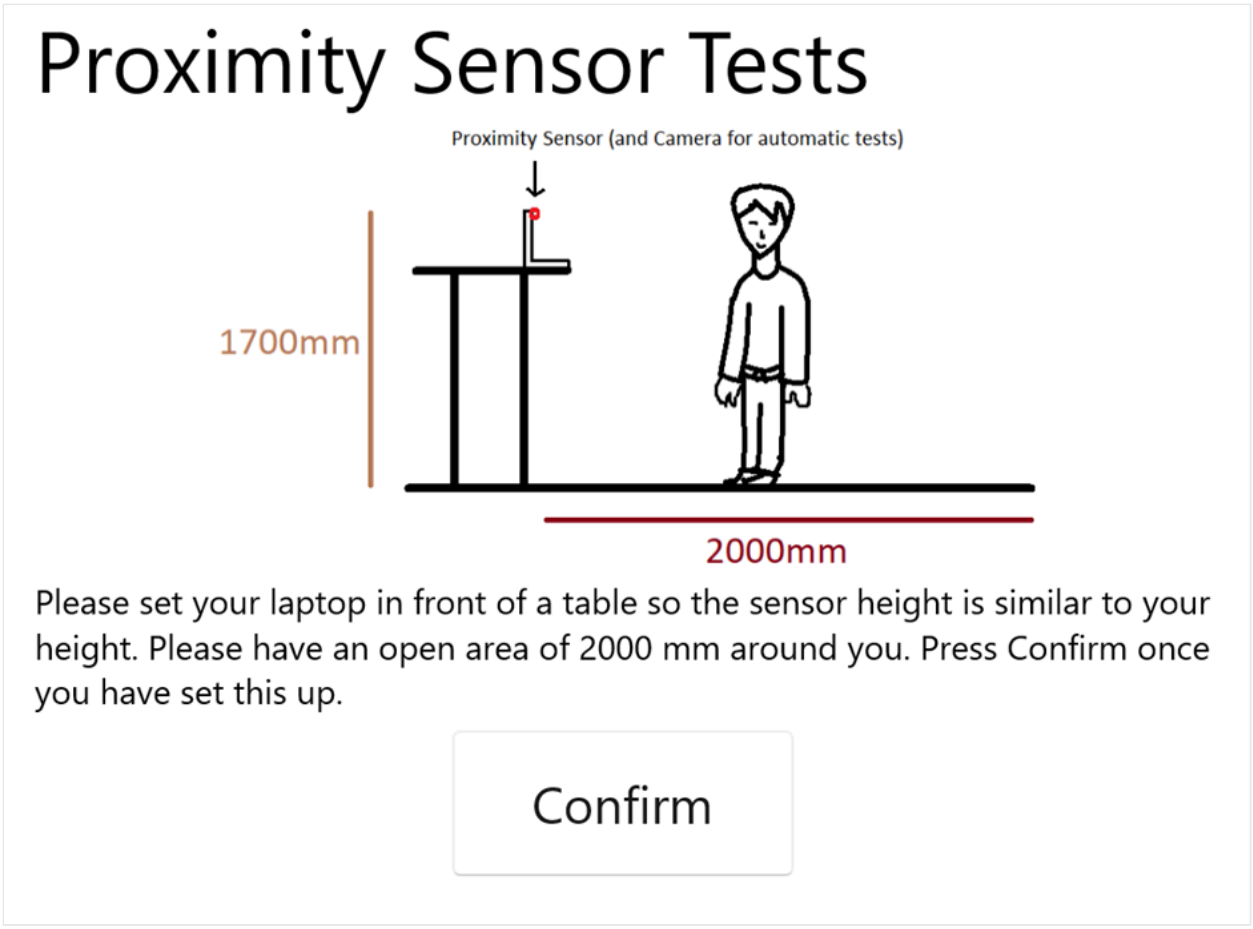
Measure	Metric	Comments
Leave matency - maximum	<15 seconds	The time between the user leaving the device (reaching the distance 45cm, 75cm, or 120cm) and the system receiving the notification of user absence.
Leave latency - average	<5 seconds	Average of the three trials.
External monitor metrics	Same as internal	This is subject to change based on experimentation and further user research studies

Accuracy - user intent recognition framework:

Expand table

Measure	Description	Target
False positives	The device screen off based on presence, but the user is not actually in front of the device.	<5%
False negatives	The user leaves the device, but the device does not turn off screen.	<5%

Directly measured: The PC placed on a standard desk, 30 inches (76.2 cm) off the floor. The user approaches the desk and chair from >1.2m away. See the following figure. This concept is only used for consistency of measurement.

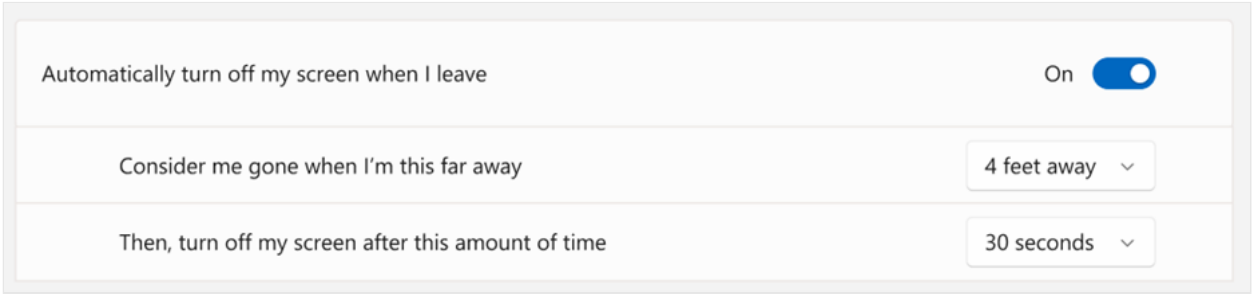

Proximity Sensor Tests
Proximity Sensor (and Camera for automatic tests)
1700mm
2000mm
Please set your laptop in front of a table so the sensor height is similar to your height. Please have an open area of 2000 mm around you. Press Confirm once you have set this up.
Confirm

Example settings

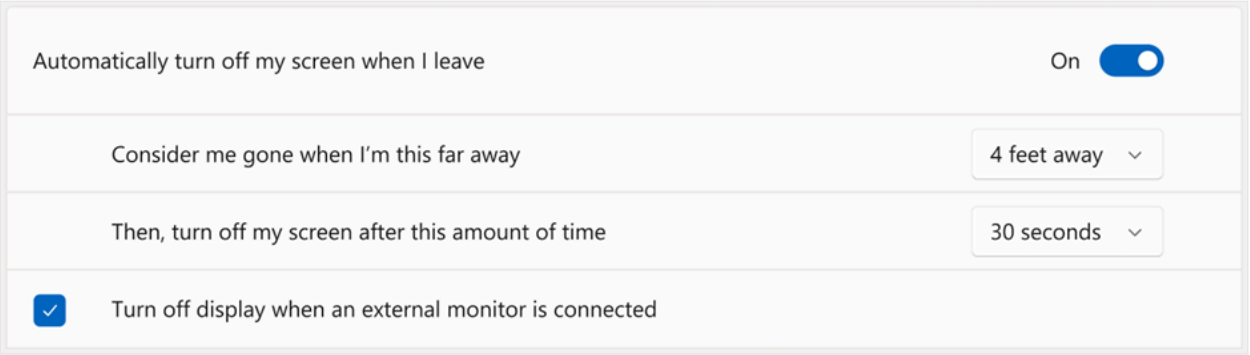
The distance setting for Lock on Leave, the distance drop-down, is only available for distance capable devices.

In the Windows 11 22H2 May 2023 update, an additional toggle is available, to enable or disable the feature when an external monitor is connected. This is checked 'on' by default.

Windows 11, version 21H2


Automatically turn off my screen when I leave
On
Consider me gone when I'm this far away
4 feet away
Then, turn off my screen after this amount of time
30 seconds

Windows 11, version 22H2 and above


Automatically turn off my screen when I leave
On
Consider me gone when I'm this far away
4 feet away
Then, turn off my screen after this amount of time
30 seconds
Turn off display when an external monitor is connected

Related topics

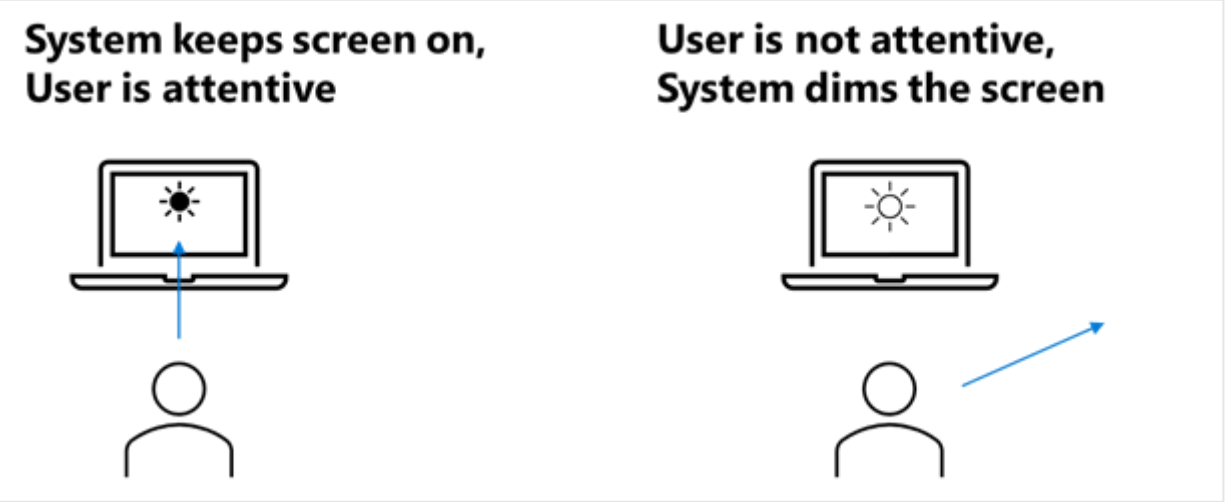
- Presence sensing device experiences
- Presence sensor hardware component guideline
- Presence sensor API documentation
- Sensors driver design guide
- Windows Hardware Compatibility Program

Adaptive Dimming

Article • 12/06/2024

This section provides an overview of the various experiences enabled by Windows 11's presence sensing features. Each experience can be enabled independently, however, Microsoft encourages the implementation of as many of these features as possible for a given hardware configuration.

Functional overview



Adaptive Dimming: When the user is sitting in front of the device with the display on, and it is detected that the user is not attentive, the operating system will dim the display. This scenario makes use of the Attention Awareness sensor which can be made available to other applications. The dim state leverages existing dim power settings.

> **Note**
>
> Adaptive Dimming is supported on Windows 11 Version 22H2 and later.

> **Tip**
>
> User engagement is generally considered based on face yaw angle. Microsoft recommends that engagement be defined with a yaw angle +-45 degrees relative to the perpendicular of the sensor (assuming user is sitting in front of sensor). No recommendations regarding face pitch or roll.

Debounce mechanisms

User is repeatedly moving in and out of the user engaged/unengaged threshold

The state machine will wait at least 5 seconds between human presence dimming to prevent the system from repeatedly switching between engaged and unengaged states. Hysteresis is provided at the OS service layer and this should be taken into account. The OS service will not take into account device-level hysteresis.

Exceptions

In any scenario that takes in a power request from the OS, dimming will not be executed. This applies to scenarios such as a video, ongoing call, and so on. This behavior will mimic the current power management schema for current "timeout-based" locks. This is not configurable to the user since these exceptions are only activated at the kernel level.

Holistic experience benchmarks

This section defines what the user should experience when using this feature, regardless of the hardware facilitating the experience.

Expand table

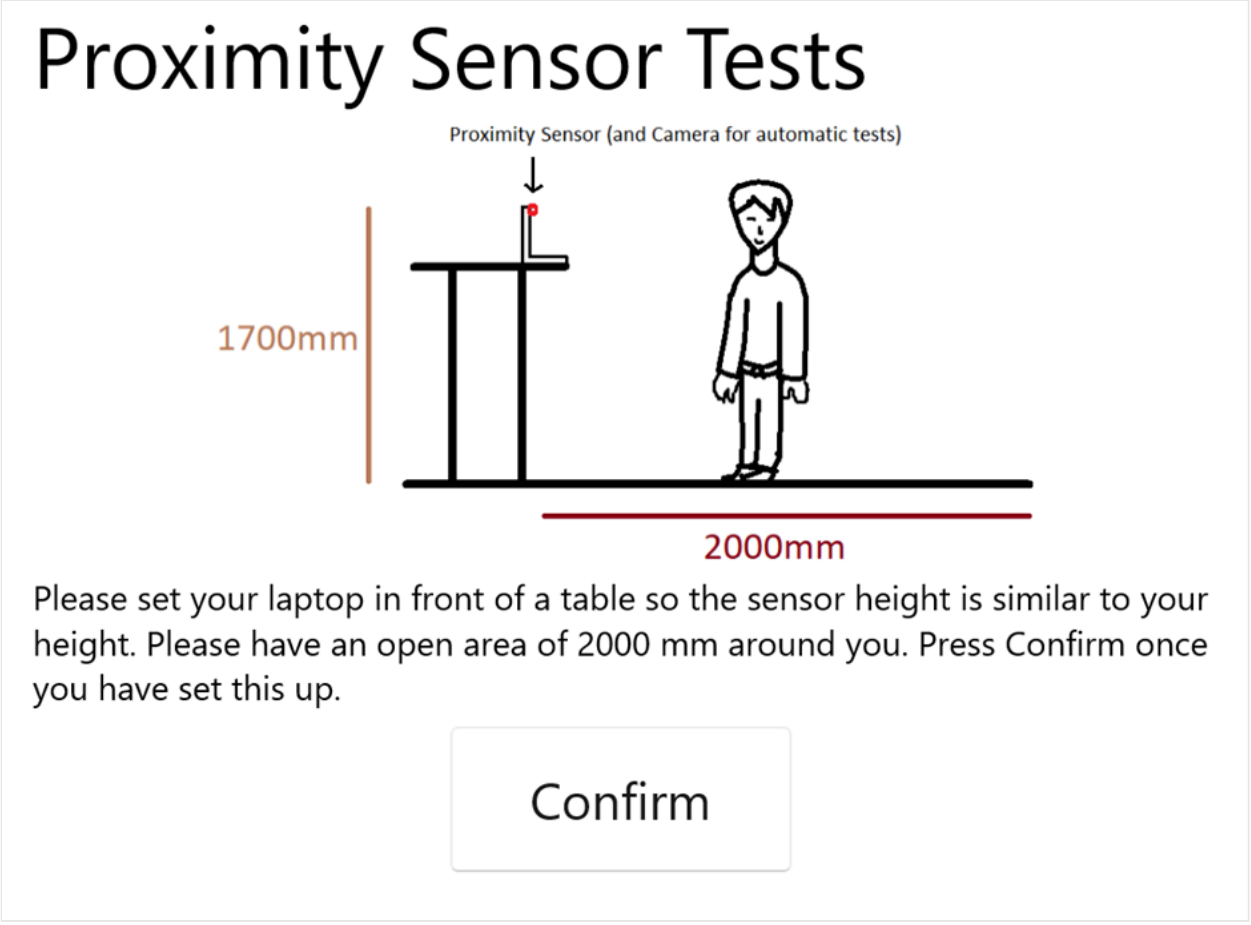
Measure	Metric	Comments
"Accuracy" - User Intent Recognition	>95% Pass Rate (determined by number of successful Test executions divided by total executions) for each scenario	
Attention Awareness Latency - Average	<2 seconds	
Attention Awareness Latency - Maximum	<5 seconds	
External monitor metrics	Same as internal	This is subject to change based on experimentation and further user research studies

Accuracy - User Intent Recognition Framework:

Expand table

Measure	Description	Target
False Positives	The awareness sensor indicates that the user is engaged but the user is not attentive to the device.	<5%
False Negatives	The awareness sensor indicates that the user is not engaged but the user is engaged/interacting with the device.	<5%

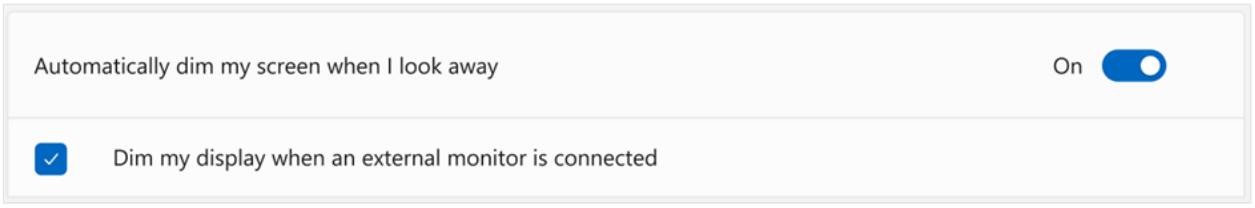
Directly measured: The PC placed on a standard desk, 30 inches (76.2 cm) off the floor. The user approaches the desk and chair from >1.2m away. See Figure below. This concept is only used for consistency of measurement.


Proximity Sensor Tests
Proximity Sensor (and Camera for automatic tests)
1700mm
2000mm
Please set your laptop in front of a table so the sensor height is similar to your height. Please have an open area of 2000 mm around you. Press Confirm once you have set this up.
Confirm

Example settings

A simple on/off toggle is made available to the user to dim the display when a user looks away. In the Windows 2023 release, an additional toggle is available, to enable/disable the feature when an external monitor is connected. This is checked **on** by default.

Windows 11, version 22H2 and above



Related topics

- Presence sensing device experiences
- Presence sensor hardware component guideline
- Presence sensor API documentation
- Sensors driver design guide
- Windows Hardware Compatibility Program

Thermal management in Windows

Article • 11/05/2020

Provides information about how Thermal management in Windows plays a key role in delivering PCs that have good performance and are safe to use — even when they are running a high-energy workload.

PCs are becoming more mobile and compact, making hardware design for thermal dissipation more challenging. At the same time, users' expectations for performance and capabilities continue to grow, increasing the heat generation of system components. Good thermal design is now more important than ever. The Windows thermal management framework offers a software solution that complements the hardware design. This infrastructure gives system designers a simple way to dictate how each component of the system reacts to thermal conditions. The control mechanisms in this framework rely on the thermal management policies defined by the system designer for a particular hardware platform. Starting with Windows 8.1, this framework is included as part of the operating system in every Windows PC. This document describes the Windows thermal management user experience and provides recommendations and guidance to system designers for how to deliver that experience using the Windows thermal management framework or through proprietary solutions. For more information, see Device-Level Thermal Management.

Principles of thermal management

Ideally, all system components in a PC should operate at peak performance while never getting too hot. In fact, no PC can operate this way. To be fully functional, all system components consume power in order to operate, and power produces heat. A PC must be prevented from overheating so that it operates reliably and is comfortable to touch. The goal of Windows thermal management is to limit the heat generated by the PC, but to do so in a way that minimally interferes with or detracts from the user experience.

Windows thermal management is based on these five principles:

- **Safety** – Regardless of workload or external conditions, the PC never becomes unsafe for a user to handle.
- **Operating range** – The PC operates over the normal operating range of ambient temperatures.
- **Full-performance experiences** – Core Windows experiences are not performance-compromised under normal operating conditions.

- **Quiet while idle** – While the PC is in a low-power state such as Modern Standby, users should not see the fan turn on under any circumstances.
- **Diagnosability** – Windows prefers any thermal mitigation measures (passive and active cooling modes) to be initiated through mechanisms that are provided by the operating system so that issues can be identified in the field and reported back via telemetry.

The following topics provide more information about the datatypes and IOCTLs used in Thermal management, along with some examples.

In this section

Topic	Description
Design Guide	This PC thermal management design guide provides information about how to determine the PC temperature values that are "too hot" and "too cold."
User Experience	From a thermal point of view, the user experience should be such that the user can operate the PC for as long as possible without interruption. And the user should only be notified of thermal problems, when the PC can no longer operate.
Examples, Requirements and Diagnostics	This topic presents examples of thermal management issues, and also discusses requirements and diagnostic methods.
Thermal Management Datatypes	This topic discusses the datatypes that are used for Thermal Management in Windows.
Thermal Management IOCTLs	This topic discusses the input/output controls (IOCTLs) that are used for Thermal Management in Windows.

Design Guide

Article • 04/27/2023

This PC thermal management design guide provides information about how to determine the PC temperature values that are "too hot" and "too cold."

Making these determinations is a key requirement for a design that delivers a good PC user experience. And also, these thresholds help to choose the first mitigating action to take for PC components that reside in multiple thermal zones.

Designing the temperature thresholds

Variables and assumptions

The following factors influence a PC's thermal behavior:

- **Hardware design**

  The importance of good hardware design cannot be over-stressed. For more information, see Hardware thermal modeling and evaluation.

- **Environment**

  These are external factors that contribute to the thermal behavior of the system. The software can only influence the environment by notifying the user that thermal constraints are an issue—for example, by displaying the thermal-failure-to-boot logo. The user must move to a different environment for these factors to change:

  - **Sunlight radiation**

    The intensity and angle of the sunlight impacting the screen or any part of the system.

  - **Ambient temperature**

    The temperature of the environment.

  - **Airflow**

    With or without air circulation. Windy or in a computer case.

  - **Humidity**

    Dry or humid.

- **Workload and power consumption**

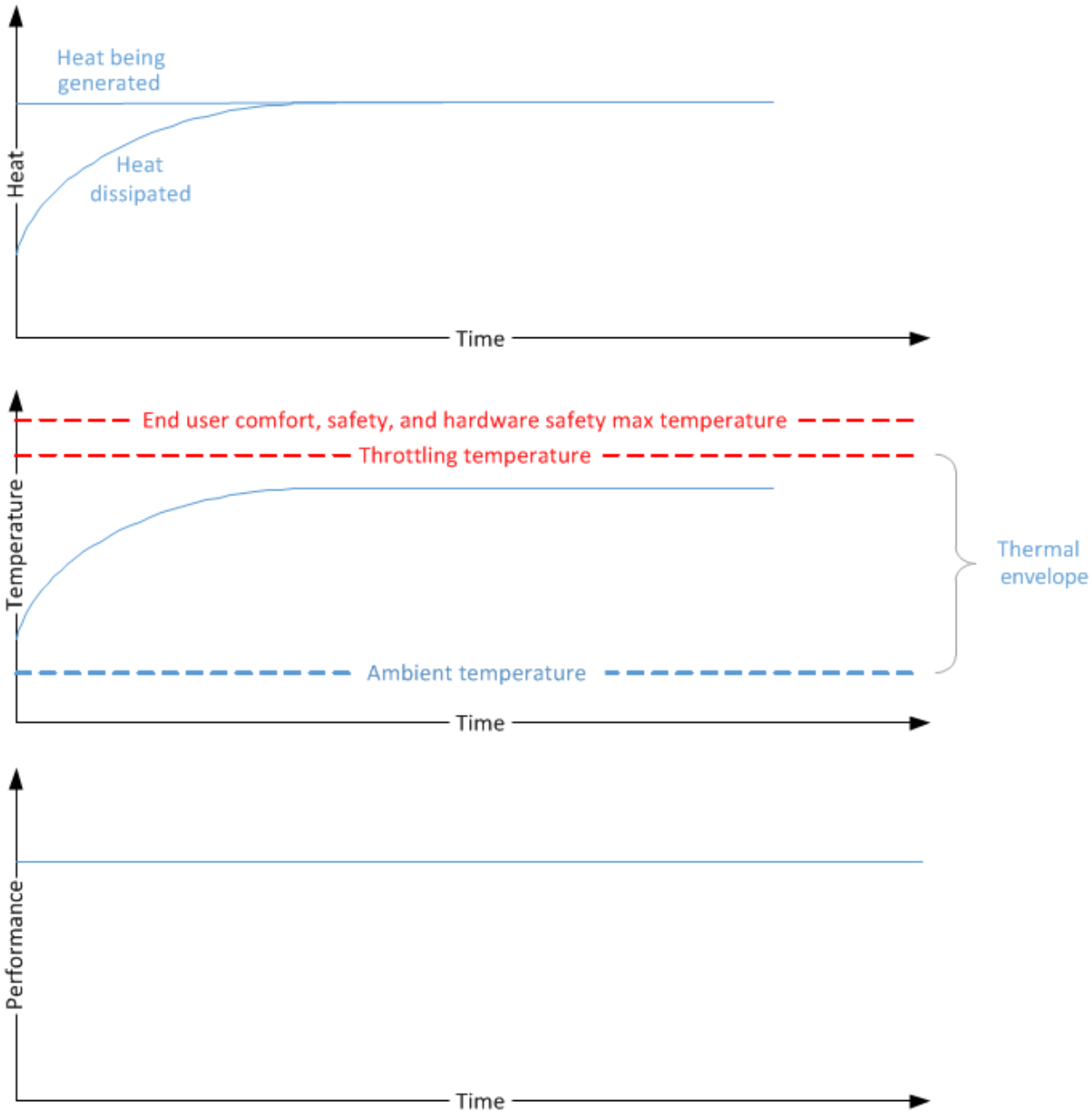
  The assumption here is that workload and power consumption are proportional to each other. In other words, the more work a PC or component does, the more power it consumes and the more heat it generates. Although this might not be true in all cases, this simplified model is more or less sufficient here. This is where software mitigations come in. By reducing the workload, less heat is generated and the PC keeps operating.

When designing and modeling the hardware, take into account the parameters mentioned in the preceding list. Please use worst-case values for the environment. The only parameter that can be directly controlled by software is the workload.

Thermal fundamentals

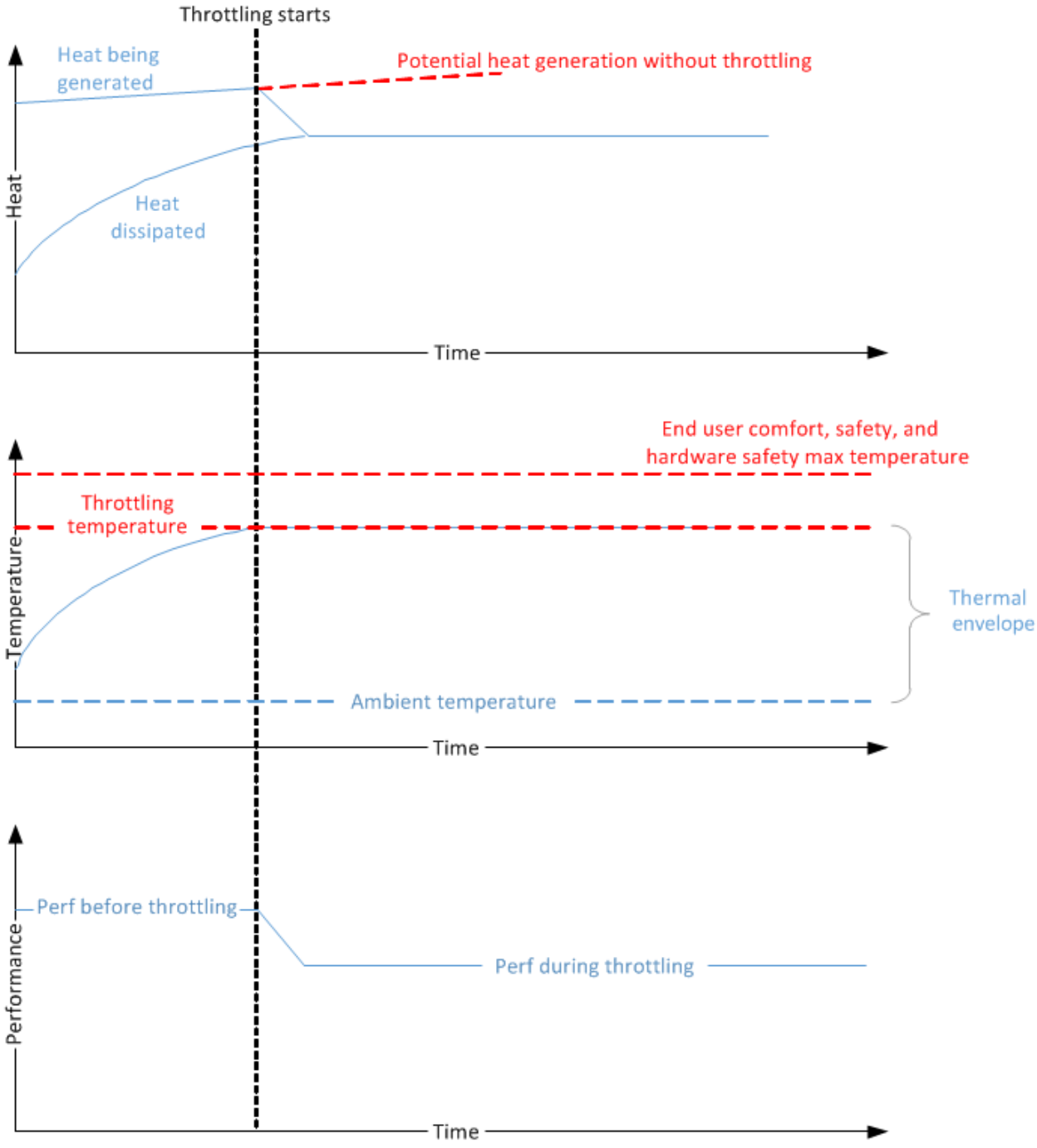
Consider a PC's thermal behavior when it is running a constant workload. As the workload starts, the PC's hardware components such as the CPU and GPU generate heat and increase the temperature. As the temperature increases relative to the ambient temperature, the PC starts to dissipate heat faster until eventually the heat generation is equal to the heat dissipation, and the temperature reaches steady state. For the entire duration of this constant workload, since there is no throttling involved, the performance and throughput are constant.

The following diagram shows the relationship between heat generation, temperature, and performance when no throttling is involved. Notice that the PC's temperature stays within the thermal envelope, as bounded by the ambient temperature and throttling temperature.



Now consider a PC's thermal behavior when it is running a different workload that is also constant but more resource intensive. As this workload executes, the heat generated is much higher than what the system is capable of dissipating to the ambient environment, and as a result the temperature rises steadily. Without passive cooling, the temperature will continue rise until eventually the system will become too hot and adversely impact end user comfort and safety. Hardware can be damaged as well at high temperatures. Thermal throttling helps ensures that the PC does not reach these high temperatures. When the temperature rises over a pre-defined throttling temperature trip point, the system starts throttling performance. As a result, the heat generation is reduced and gradually—after the heat generation and dissipation equalize—the system temperature reaches the steady state.

The following diagram shows the relationship between heat generation, temperature, and performance when performance is throttled to reduce heat generation.



In both cases shown in the preceding diagrams, the workloads must operate within a thermal envelope to ensure that the system temperature does not exceed safe limits. The envelope starts with the ambient temperature and ends with the throttling temperature. Also in both cases, the heat generation and dissipation eventually reach a balanced state, and the system temperature is stabilized.

Defining a thermal envelope

A well-designed PC should have as large of a thermal envelope as possible, providing users a long-lasting, mitigation-free experience. As shown in the preceding diagrams, the thermal envelope has a lower bound determined by the ambient temperature. It is bounded above by the throttling temperature. While the ambient temperature is not a variable that system designers can control, the upper bound can be pushed higher by

good hardware design. For more information, see Hardware thermal modeling and evaluation. But even assuming that hardware is not the major limitation, other important factors must be considered when defining the thermal envelope.

The desired operating range should be as large as possible without encroaching on these limiting factors:

- **Safety**

  Temperature of the system must first ensure that users do not suffer any injuries from using the system. This requirement applies mostly to the skin temperature sensor.

- **Hardware protection**

  Temperature should prevent system components from "melting" or otherwise causing damage due to heat. This requirement applies mostly to component temperature sensors—for example, a sensor that sits on top of the processor.

- **Government regulation**

  All systems should meet applicable industry standards (such as IEC 62368) for consumer electronics safety.

Hardware thermal modeling and evaluation

Software as complement to hardware design

When designing hardware, it's of key importance to keep thermal management in mind. Selecting low-power parts, placing hot components far from one another, and incorporating thermal insulation are only a few examples of how hardware can greatly improve the thermal experience. These methods cannot be replaced in software. As such, the software solution only serves as a complement to the hardware design in the overall thermal experience.

Hardware goal

Typical workloads should not need any form of software thermal mitigation to run. The hardware thermal design should be able to disperse heat for these workloads by itself.

Modeling

Modeling is an iterative process to achieve the hardware goal previously described. In this process, do not factor in any software mitigations. Rely solely on the hardware capabilities and adjust as necessary.

1. Define a typical workload. Depending on the platform of the system, from phone to server, each system should have a standard set of typical workloads. These should not be intense workloads such as HD video conferencing, but rather a more average workload like browsing the web or listening to music.

2. Model system and individual component power consumption on typical workloads since heat will not be spread across the chassis uniformly. Pay particular attention to the components that consume most power, such as the processor.

3. Based on the workload power consumption estimation, model the rise in component and skin temperature over time.

4. Adjust the system's mechanical design to ensure that the component temperature is within the hardware safety limit and user comfort limit over all ambient temperatures. Some methods for addressing mechanical design issues are:
   a. Improve the heat dissipation capability by adding better heat-conducting materials.
   b. Increase the delta temperature between the skin and the internal temperature by adding isolation layers.

5. Repeat steps 1 through 4 until satisfied.

6. Build the hardware and evaluate.

Evaluation

For every hardware revision, a temperature and power measurement against representative workloads must be performed to evaluate the thermal behavior, and the mechanical design should be modified as needed.

The following steps are recommended to perform such thermal evaluation:

1. Define a thermal measurement test matrix:
   a. The matrix should cover both normal and maximum ambient temperature.
   b. The matrix should include all typical workloads identified as part of the thermal modeling, and for every workload, data should be captured for at least one hour.
   c. The matrix should be executed in multiple PCs multiple times to generate consistent results.

2. For every workload defined in the test matrix, capture the following data:
   a. System and component power consumption data.
   b. Ambient, internal component, and skin temperature data.

c. Software traces to detect thermal throttling, performance metrics, and processor utilization.

The skin and component temperature data directly indicates the maximum skin temperature that the system might reach and whether this temperature is acceptable. Consider how much thermal headroom the system might have before critical shutdown. The performance metrics data will help determine whether the system is delivering the required performance. The trace data recorded by the thermal throttling software shows the thermal throttling percentage. The power consumption data and the CPU utilization data can help to determine what factors influence the temperature changes.

The following table provides an example of such data collection for a PC with the following configuration:

Configuration

- Machine name: Thermal-Test-1
- Average ambient temperature: 23°C
- Thermal zone trip point (_PSV): 80°C

Category	**Subcategory**	**Video streaming**	**Casual gaming**	**Fishbowl**	**3D gaming**	**TDP**
Workload setup		Full-screen 720p H.264 video streaming through Wi-Fi	Game name CPU utilization	Classic IE with 100 fish	Game name CPU utilization	CPU and GPU 100%
Power consumption	System power					
	SoC power					
	Display power					
	Backlight power					
Temperature	Max SoC temperature					
	Max skin temperature					

Category	Subcategory	Video streaming	Casual gaming	Fishbowl	3D gaming	TDP
Performance metrics		Average frame rate	Average frame rate	Average frame rate	Average frame rate	

Windows thermal management framework

The Windows thermal management framework provides a comprehensive solution for software thermal management. The following examples show how to implement thermal management. With proper thermal modeling, validation, and effective thermal mechanical design, all systems should be able to deliver a smooth user experience on most workloads on most targeted ambient temperatures. Where thermal mitigation is required, Windows provides an effective and extensible thermal management architecture.

Windows thermal management supports both active and passive cooling. With active cooling, fans turn on to circulate air and help cool the system. With passive cooling, devices reduce device performance in response to excessive thermal conditions. Reducing performance lowers the power consumption, and thus generates less heat.

The Windows thermal management framework relies on the policies specified by system designers to enforce thermal management on the system. At a high level, designers must specify how the reading obtained from each thermal sensor is affected by each component. Without these specifications, Windows cannot thermally manage the system. Thus it is the responsibility of each system designer to characterize their system thermally in order to achieve a good thermal design.

Although systems are not required to use the Windows thermal management framework, it is the recommended solution because of its tight integration with the operating system. However, regardless of the thermal management solution used, all Modern Standby PCs must adhere to the HCK requirements for diagnostic purposes.

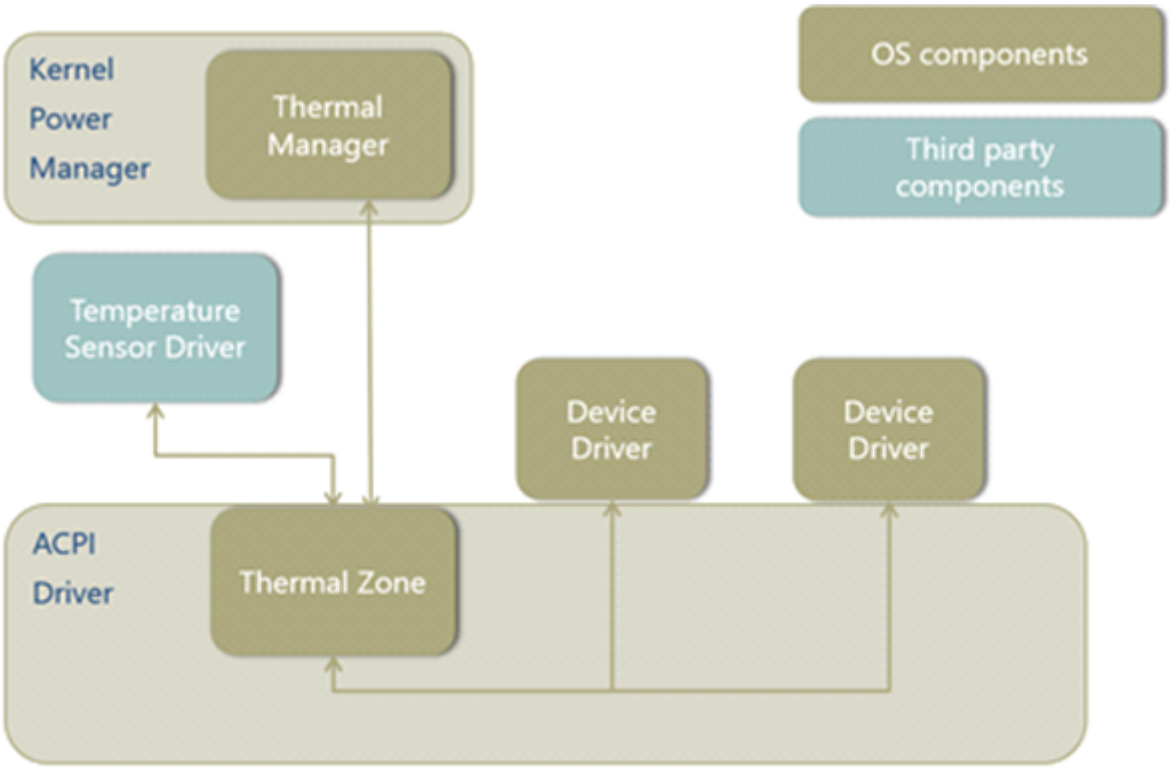
Thermal management architecture

The Windows thermal management architecture is based on the ACPI concept of *thermal zones*. The ACPI thermal zone model requires cooperation between the firmware, operating system, and drivers. This model abstracts the sensors and cooling devices from the central thermal management component through well-defined interfaces. The thermal management enhancements are based on Chapter 11 of the ACPI 5.0 specification. The Windows thermal management framework implements a subset of the capabilities described in this chapter.

The essential components of the model are:

- Platform **thermal zone** definitions described to Windows via the firmware. A thermal zone is an abstract entity that has an associated temperature value. The operating system monitors this temperature so that it can apply thermal mitigation to devices within the zone. The zone contains a set of functional devices that generate heat, and a subset of devices whose heat generation can be controlled by adjusting performance.
- A **temperature sensor** that represents the region's temperature. The sensor must implement the Read Temperature interface to retrieve the temperature of a zone from firmware or a Windows temperature driver.
- A **thermal cooling interface** to enable drivers for devices within thermal zones to implement passive-cooling actions. Each managed driver must have the cooling interface to participate in Windows thermal infrastructure.
- A centralized **thermal manager** that orchestrates cooling by interpreting the thermal zone definitions and invoking the interfaces at the required times. The thermal manager is implemented in the Windows kernel and requires no work from system designers.

The following block diagram is an overview of the Windows thermal management architecture. The main components are the thermal manager, thermal zone, managed drivers, and temperature sensor. The thermal zone dictates the throttling behavior of its managed devices based on constraints provided by the thermal manager. The algorithm used by the thermal manager uses the temperature reading of the sensor for the thermal zone as its input parameter.



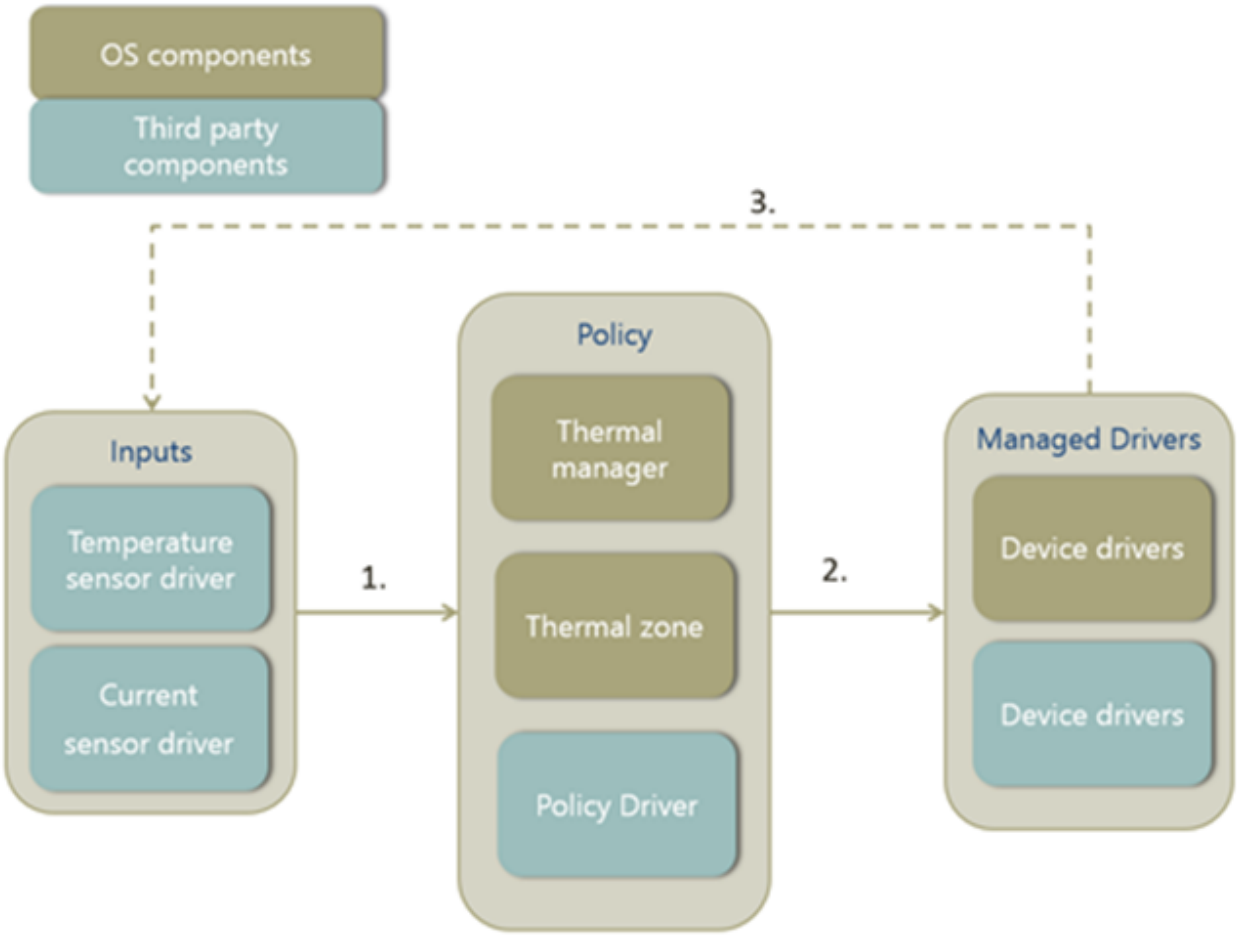
The thermal zones in the system must be described in ACPI firmware and exposed to the thermal manager through ACPI. With the configuration information, the thermal manager knows how many thermal zones need to be managed, when to start throttling each thermal zone, and which devices are a part of the zone. To monitor temperature, the system designer provides support in ACPI firmware for thermal notifications.

When the thermal manager is notified of a thermal event in an enumerated zone, it will start to periodically evaluate the temperature of the zone and determine the thermal throttling performance percentage to apply to devices in the thermal zone. This evaluation is done by the thermal throttling algorithm that is outlined in the ACPI specification. Then the thermal manager notifies all the devices in the zone to throttle performance by a specific percentage and the device drivers translate the throttling percentage to a device-class-specific action to reduce performance. The periodic evaluation and throttling will stop when the thermal zone temperature falls below the throttling threshold temperature and no more throttling is required.

Feedback loop

Another way to think of the thermal management architecture is in terms of inputs, policy director, and managed devices. In the following block diagram, the inputs to the feedback loop are the temperature and electrical current. These inputs are used to determine the thermal policy to implement. The thermal manager relies on the temperature input exclusively, and the policy driver may use whatever input it desires. The thermal zone then applies that policy to its managed drivers. After the policy is applied, the sensors will update their values and close the loop.

The following block diagram shows the three stages of the thermal response model. Inputs from temperature and electrical current sensors provide information to help determine what thermal policy to apply. This policy gets applied to the managed drivers, which then affects the readings on the sensors. The process repeats in a feedback loop.



Responsibilities of system implementers

As mentioned above, a number of components are required to have a complete Windows thermal solution. The architecture of the thermal framework is specifically designed so that the responsibilities of the hardware vendor and system integrator can be separated.

Required components for partners to implement are:

- **Sensors**

  The temperature sensor drivers should be provided by the hardware vendor. Given that temperature sensors do not need knowledge of the thermal zones that rely on them, their functionalities should be standard across different platform designs. Custom sensors for policy drivers, such as current sensors, are also the responsibility of the hardware vendor.

- **Thermal zones**

  The thermal zones can be defined by the hardware vendor and/or system integrator. All systems must have at least one thermal zone that implements a critical shutdown temperature (and hibernate temperature, if supported), which can be done by the hardware vendor or system integrator. However, for other thermal zones that monitor specific devices or the skin temperature for mitigation,

it's common for the system integrator to have more specific knowledge of the thermal behavior of the system. Thus, these thermal zones are usually implemented by the system integrator.

- **Thermal cooling interface for device drivers**

  The developer who writes the driver for the device that is to be thermally managed should also be the one to implement the cooling interface. The device driver uses this interface to opt into the thermal management framework. Device drivers have specific knowledge of the capabilities of their devices. These same drivers must implement the thermal cooling interface so that it can properly interpret the actions requested by the thermal zone.

Sensors

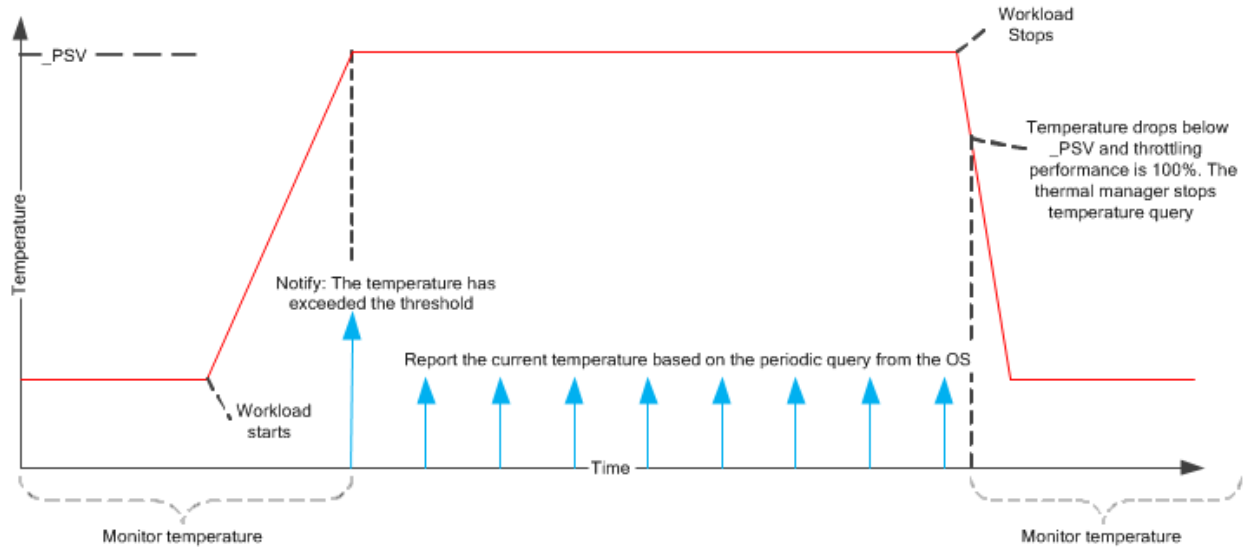
Sensors provide inputs to determine what the thermal policy should be. Windows supports only temperature sensors as inputs to the thermal manager. However, a policy driver can additionally take inputs from custom drivers, such as a current sensor driver.

Temperature sensor

The temperature sensor provides the following modes of functionality:

- Continuously monitors temperature changes without the involvement of the thermal manager or thermal zone.
- Notifies the thermal manager when the temperature crosses the threshold defined by _PSV, _HOT, or _CRT.
- Responds to temperature queries and returns the current temperature value.

The following diagram shows how the temperature monitoring functions during throttling. The ACPI firmware or the temperature sensor driver should notify the thermal manager when the temperature reaches a predefined threshold such as _PSV, _HOT or _CRT, and then respond to the periodic queries from the thermal manager for the current temperature. The period of the temperature query is defined by _TSP.


_PSV
Temperature
Workload Stops
Temperature drops below _PSV and throttling performance is 100%. The thermal manager stops temperature query
Notify: The temperature has exceeded the threshold
Report the current temperature based on the periodic query from the OS
Workload starts
Time
Monitor temperature
Monitor temperature

To make sure that the thermal manager will be always notified when the temperature exceeds the threshold, the temperature sensor interrupt should be always wake-able (even when the system is in Modern Standby and the SoC is in a low-power state). If the temperature sensor interrupt is not always wake-able, then at least the interrupt should be configured as level-triggered to avoid potential interrupt loses.

A thermal sensor can be used by multiple thermal zones, although there can be no more than one thermal sensor in a thermal zone.

We recommend that the sensor hardware to be accurate to +/- 2°C.

The temperature reported by _TMP or a temperature sensor driver may be the actual value of a sensor, or an extrapolated value based on several sensors.

This is usually provided by the hardware vendor. Windows supports two implementations for monitoring temperature:

- Temperature sensor driver
- ACPI-based

Implementation 1: Temperature sensor driver

The *temperature sensor driver* simply reports the temperature of the sensor. The ACPI driver will issue one outstanding IOCTL with the sensor driver to detect a crossing of one of the trip points. In addition, ACPI may issue one IOCTL with no time-out to read the current temperature. The sensor driver should handle cancellation of the read IOCTL, if it is canceled while waiting for the time-out to expire. The temperature sensor must implement the following interface:

```
typedef struct _THERMAL_WAIT_READ {
    ULONG Timeout;
    ULONG LowTemperature;
    ULONG HighTemperature;
} THERMAL_WAIT_READ, *PTHERMAL_WAIT_READ;

#define IOCTL_THERMAL_READ_TEMPERATURE\
        CTL_CODE(FILE_DEVICE_BATTERY, 0x24, METHOD_BUFFERED,
FILE_READ_ACCESS)
```

The following table describes the input and output parameters for the temperature-reading interface.

Parameter	Description
Timeout	The time to wait before returning temperature data, in milliseconds. 0 indicates the temperature should be read immediately, with no wait. -1 indicates the read should not time out.
LowTemperature	The lower threshold for returning the new temperature. As soon as the temperature drops below the low temperature threshold, the driver should complete the IOCTL. If the temperature is already below the low temperature, the IOCTL should be completed immediately.
HighTemperature	The upper threshold for returning the new temperature. As soon as the temperature rises above the high temperature threshold, the driver should complete the IOCTL. If the temperature is already above the high temperature, the IOCTL should be completed immediately.
IOCTL return value	A ULONG-sized output buffer that will return the current temperature, in tenths of a degree Kelvin.

One temperature sensor can be used by one thermal zone or multiple thermal zones. To specify which temperature sensor should be used for a thermal zone, the thermal _DSM should be specified on the thermal zone, and should implement function 2. (For backwards compatibility, the temperature sensor driver can be loaded on top of the thermal zone stack. However, the preferred implementation is to have the sensor driver separate from the thermal zone stack.) This _DSM is defined as follows:

Arguments *Arg0:* UUID = 14d399cd-7a27-4b18-8fb4-7cb7b9f4e500 *Arg1:* Revision = 0 *Arg2:* Function = 2 *Arg3:* An empty package **Return** A single Reference to the device that will return the temperature of the thermal zone.

The thermal zone must also specify a dependency on the temperature sensor device with _DEP. Here's a simple example for _DSM implementation of a sensor device.

```
Device(\_SB.TSEN) {
    Name(_HID, "FBKM0001")     // temperature sensor device
}

ThermalZone(\_TZ.TZ01) {
    Method(_DSM, 0x4, NotSerialized){
        Switch (ToBuffer(Arg0)) {
            Case (ToUUID("14d399cd-7a27-4b18-8fb4-7cb7b9f4e500")) {
                Switch (ToInteger(Arg2)) {
                    Case(0) {
                        // _DSM functions 0 and 2 (bits 0 and 2) are
supported
                        Return (Buffer() {0x5})
                    }
                    Case (2) {
                        Return ("\_SB.TSEN")
                    }
                }
            }
        }
    }

    Method(_DEP) {
        Return (Package() {\_SB.TSEN})
    }

    // Other thermal methods: _PSV, etc.

}
```

For more information, see Device-Specific Method for Microsoft thermal extensions.

Implementation 2: ACPI-based

The ACPI firmware should support _TMP and Notify 0x80, as defined in the ACPI specification. The advantage of this approach is that it does not require any additional drivers be installed on the system. However, this approach is limited to interacting with resources that are accessible through ACPI operation regions.

Thermal policy control

Thermal zone

A thermal zone is an individual thermal throttling entity. It has its own thermal throttling characteristics, such as trip points, throttling sample rate, and throttling equation constants. One thermal zone might include multiple thermal throttling devices, each of which can contribute to temperature increases in the thermal zone. All devices in the same thermal zone must follow the same constraints applied to the thermal zone.

To make sure thermal zones and their parameters are defined accurately, system designers should:

- Identify hot spots in the system's chassis, and determine how these hot spots dissipate heat to the temperature sensors. (Ideally, thermal sensors are sitting at the hot spots on the system, except for skin temperature sensors.)
- Characterize the system's temperature relationship. Map the temperature sensor readings to the component temperature and the skin temperature.

Overthrottle threshold

Starting with Windows 10, a new feature called a *thermal zone state*, has been introduced into Windows thermal management, along with one state: *overthrottled*. When a thermal zone exceeds the device's designed throttle level, the thermal manager will notify the operating system components that the system is overthrottled. This notification allows the system to reduce workload to improve the thermal state.

The thermal manager maintains a global count of the number of thermal zones that are in the overthrottled state. When the count increases above zero (0), the thermal manager sends out a Windows Notification Facility (WNF) notification to the system, indicating that it is overthrottled. When the number of overthrottled zones returns to zero (0), the thermal manager sends out another WNF notification to the system, indicating there are no overthrottled zones.

To specify the overthrottle threshold for a thermal zone, the thermal _DSM should be specified on the thermal zone, with function 3 implemented. This _DSM is defined as follows:

Arguments *Arg0:* UUID = 14d399cd-7a27-4b18-8fb4-7cb7b9f4e500 *Arg1:* Revision = 0 *Arg2:* Function = 3 *Arg3:* An empty package **Return** An Integer value with the current overthrottle threshold, expressed as a percentage.

Here's an example _DSM which indicates that the zone is overthrottled, at throttle levels 0% through 49%.

```
ThermalZone (TZ4) {
   Name (_HID, "QCOM24AE")
   Name (_UID, 0)
   Name(_TZD, Package (){\_SB.CPU4, \_SB.CPU5, \_SB.CPU6, \_SB.CPU7})
   Method(_PSV) { Return (3530) }
      Name(_TC1, 1)
      Name(_TC2, 1)
      Name(_TSP, 1)
      Name(_TZP, 0)
      // _DSM - Device Specific Method
      // Arg0: UUID Unique function identifier
      // Arg1: Integer Revision Level
      // Arg2: Integer Function Index (0 = Return Supported Functions)
      // Arg3: Package Parameters
      Method(_DSM, 0x4, NotSerialized) {
         Switch (ToBuffer(Arg0)) {
            Case (ToUUID("14d399cd-7a27-4b18-8fb4-7cb7b9f4e500")) {
               Switch (ToInteger(Arg2)) {
                  Case(0) {
                     // _DSM functions 0 and 3 (bits 0 and 3) are supported
                     Return (Buffer() {0x9})
                  }
                  Case (3) {
                     Return (50)  // overthrottled below 50%
                  }
               }
            }
         }
      }
} // end of TZ4
```

The overthrottle threshold will be re-read when a Notify(0x81) is received in reference to the thermal zone.

Implementing ACPI objects

The following table lists all ACPI objects that need to be implemented in ACPI firmware to define a thermal zone.

Category	**Control method**
Identify the devices contained within the zone	**_TZD** Lists the devices in the thermal zone. **_PSL** (Optional) Lists the processors in the thermal zone. Processors may be included in _TZD instead—Windows supports both implementations.

Category	Control method
Specify thermal thresholds at which actions must be taken	**_PSV** The temperature at which to start throttling. For more information, see Defining a thermal envelope. **_HOT** (Optional) The temperature at which the operating system hibernates the system. For systems that do not support hibernate, the operating system will initiate critical shutdown. This is highly recommended for at least one thermal zone for all x86/x64 systems because of the benefit of hibernate to save user data. **_CRT** The temperature at which the operating system initiates critical shutdown. No user-mode notification. At least one thermal zone on a system must have _CRT specified. If not, the system does not go through the shutdown path when the system reaches critical temperature. Instead, the firmware fail-safe trip point is reached and the power is likely pulled from under the operating system.
Describe passive cooling behavior	**_TC1** Control how aggressively the thermal manager applies thermal throttling performance against temperature change. **_TC2** Control how aggressively the thermal manager applies thermal throttling performance against temperature delta between the current temperature and _PSV. **_TSP** Appropriate temperature sampling interval for the zone in tenths of a second. The thermal manager uses this interval to determine how often it should evaluate the thermal throttling performance. Must be greater than zero. For more information, see Thermal throttling algorithm.
Describe active cooling behavior	**_ACx** (Optional) The temperature at which to turn on fan. Must be in order of greatest to least, with _AC0 being greatest. **_ALx** List of active cooling devices.

Category	Control method
Set active / passive cooling policy	**_SCP** (Optional) To set the user's preferred cooling policy, if both active and passive cooling are supported by a zone.
Minimum Throttle Limit	**_DSM** Use UUID: 14d399cd-7a27-4b18-8fb4-7cb7b9f4e500 to set minimum throttle limit. Note that this is custom to Windows thermal framework and is not defined in ACPI. For more information, see Minimum throttle limit.
Report temperature	**_TMP** Read the temperature of the thermal zone. **_HID** A vendor-unique hardware identifier to load the Windows temperature driver. **_DTI** (Optional) For notifying the platform firmware that one of the zone's thermal thresholds has been exceeded. This method enables the firmware to implement hysteresis by changing the zone's thresholds. **_NTT** (Optional)For specifying significant temperature changes that the platform firmware should also be notified of via _DTI.
Notify the thermal manager	**Notify 0x80** Notifies the operating system that the threshold (_PSV) has been exceeded. The Windows thermal manager will commence thermal control. **Notify 0x81** (Optional) Notifies the operating system that the threshold values of the zone have changed. The Windows thermal manager will update itself to use the new values. This method is typically used to implement hysteresis.
Specify temperature sensor device	**_DSM** Use UUID: 14d399cd-7a27-4b18-8fb4-7cb7b9f4e500. For more information, see Temperature sensor. **_DEP** Load a device that the temperature sensor refers to.

Minimum throttle limit

The minimum throttle limit is a Microsoft thermal extension _DSM method that creates a lower bound for _PSV requested of a throttled device. In other words, it determines how much a thermal zone limits the performance of devices it controls. If present, the minimum throttle _DSM will be read at boot and any time the thermal zone receives ACPI Notify (0x81). At every iteration of the passive cooling algorithm, the following is used to compute the change in performance limit (DP) that the thermal manager applies to devices in the zone:

DP [%] = _TC1 × (T_n – T_{n-1}) + _TC2 × (T_n – T_t) We then use the following to compute the actual performance limit:

P_n = P_{n-1} – DP With the minimum throttle limit (MTL) implemented, this second equation changes to:

P_n = max(P_{n-1} – DP, MTL) To specify the minimum throttle limit for a thermal zone, the thermal _DSM should be specified on the thermal zone, with function 1 implemented. This _DSM is defined as follows:

Arguments *Arg0:* UUID = 14d399cd-7a27-4b18-8fb4-7cb7b9f4e500 *Arg1:* Revision = 0 *Arg2:* Function = 1 *Arg3:* An empty package **Return** An Integer value with the current minimum throttle limit, expressed as a percentage.

Here's a simple example for _DSM limiting throttle no less than 50 percent.

```
ThermalZone(\_TZ.TZ01) {
    Method(_DSM, 0x4, NotSerialized) {
        Switch (ToBuffer(Arg0)) {
            Case (ToUUID("14d399cd-7a27-4b18-8fb4-7cb7b9f4e500")) {
                Switch (ToInteger(Arg2)) {
                    Case(0) {
                        // _DSM functions 0 and 1 (bits 0 and 1) are
supported
                        Return (Buffer() {0x3})
                    }
                    Case (1) {
                        Return (50)
                    }
                }
            }
        }
    }
```

Thermal manager in kernel

The Windows thermal manager is implemented as part of the Windows kernel. It monitors the temperature of all thermal zones and applies thermal throttling as appropriate.

Every time the thermal manager queries the ACPI driver for the current temperature, it will perform a calculation on how much thermal throttling performance percentage it should apply to all thermal throttling devices in the thermal zone. The thermal limit is evaluated and applied when the passive cooling threshold (_PSV) is exceeded, and at every temperature sampling interval (_TSP) until the temperature is cooled below it and the thermal limit returns to 100 percent. The hardware must detect when the _PSV has been exceeded and must signal that via a hardware ACPI event notification.

Thermal throttling algorithm

The thermal throttling algorithm uses the following equation, which is defined in the ACPI specification:

DP [%] = _TC1 × (T_n – T_{n-1}) + _TC2 × (T_n – T_t) *where*

T_n = current temperature reading of the temperature sensor in the thermal zone in tenths of a degree Kelvin. T_{n-1} = temperature from the previous reading. T_t = target temperature in tenths of a degree Kelvin (_PSV). DP = performance change required. This is a linear value between 0 (fully throttled) and 100 percent (unthrottled), which is to be applied to each cooling device in the zone. This equation describes the feedback loop between the temperature changes and the throttling performance. In each iteration of the loop, the temperature delta between the current and previous temperature readings requires performance P to decrease by DP percent. The DP value is the amount of *thermal throttling* to apply. Many cooling devices do not have a linear thermal response to cooling mitigations. In these devices, the thermal limit is a hint to the device to indicate how much cooling is required. Each cooling device will have its own mapping of this linear value to device-specific thermal mitigations. Throttling device performance reduces power consumption and heat generation, which causes the temperature to decrease.

The two constants, _TC1 and _TC2, control how aggressively thermal throttling is applied in this feedback loop. The bigger _TC1 is, the more aggressively thermal throttling is used to maintain a stable temperature. The bigger _TC2 is, the more aggressively thermal throttling is used to push the temperature near the trip point.

The following table provides an example of how the thermal manager calculates and applies the DP. This example uses the following parameter values:

Configuration

- _PSV = 325°K
- _TC1 = 2
- _TC2 = 3
- _TSP = 5000 milliseconds
- Assume the temperature continuously rises by 1 degree every 5 seconds.

The rightmost column in the following table (labeled P) indicates the throttled performance level that results from enforcing the constraints specified by these parameters.

Iteration	Time	T_n	DP	P
1	0 seconds	326°K	= 2 × 1 + 3 × 1 = 5%	95%
2	5 seconds	327°K	= 2 × 1 + 3 × 2 = 8%	87%
3	10 seconds	328°K	= 2 × 1 + 3 × 3 = 11%	76%
4	15 seconds	320°K	= 2 × 1 + 3 × 4 = 14%	62%
5	20 seconds	330°K	= 2 × 1 + 3 × 5 = 17%	45%
...				

Policy driver

By default, the algorithm used to determine the throttling percentage as dictated by ACPI specs is used for all thermal zones. As previously described, this algorithm is a based solely on the temperature sensor attached to the thermal zone, which can be limiting.

If a different algorithm is preferred, the system designer can implemented a policy driver to embody the preferred algorithm. A policy driver sits on top of the thermal zone stack for the zone it controls. For this zone, the policy driver's algorithm can be used in place of the ACPI algorithm in the thermal manager. The policy driver's algorithm has the capability to consider any information it can access as input. The policy decisions made by the driver are passed to the thermal manager, which logs the information and passes it to the thermal zone to carry out.

Using a policy driver for a thermal zone means that the policy driver—not ACPI and not the operating system—is solely responsible for deciding when to throttle a zone and by how much.

If a policy driver is present, all trip points, all thermal constants, the minimum throttle limit, and so on, are completely ignored. The operating system has no insight into why the thermal zone is set at its current throttling level. Some benefits come with using a

policy driver instead of a propriety solution. A policy driver leverages the built-in process of throttling devices. Anything that a thermal zone can do to provide thermal mitigation can be done by a policy driver. In addition, diagnostics for Windows thermal management are automatically inherited.

The thermal policy interface has been updated to include a new parameter to indicate whether or not the zone is overthrottled. This parameter is initialized to FALSE. Old policy drivers will be unaware of the new parameter, and new drivers will know that the new parameter is supported, when they detect a policy version of '2'.

```
#define TZ_ACTIVATION_REASON_THERMAL      0x00000001
#define TZ_ACTIVATION_REASON_CURRENT      0x00000002

#define THERMAL_POLICY_VERSION_1          1
#define THERMAL_POLICY_VERSION_2          2

typedef struct _THERMAL_POLICY {
    ULONG           Version;
    BOOLEAN         WaitForUpdate;
    BOOLEAN         Hibernate;
    BOOLEAN         Critical;
    ULONG           ActivationReasons;
    ULONG           PassiveLimit;
    ULONG           ActiveLevel;
    BOOLEAN         OverThrottled;
} THERMAL_POLICY, *PTHERMAL_POLICY;
```

The policy driver can indicate that the thermal zone is overthrottled, by completing the policy IOCTL with the *OverThrottled* parameter set to TRUE. When thermal conditions improve, the thermal policy driver can then complete the IOCTL with *OverThrottled* reset to FALSE, to indicate that the thermal zone has recovered. Windows will not require the policy driver to be throttling when the overthrottle flag is set.

Thermally managed devices

Thermal zones control the thermal behavior of managed devices through their kernel mode drivers. One thermal throttling device might reside in multiple thermal zones. When multiple thermal zones requests different thermal throttling performance percentages, the thermal manager picks the minimal thermal throttling performance percentage to apply to the thermal throttling device.

Many cooling devices do not have linear thermal response to cooling mitigations. In these cases, the thermal limit is a hint to the device of the degree of cooling required.

Each cooling device will have its own mapping of this linear value to its specific thermal mitigations.

As each device driver is loaded, ACPI will query for the Thermal Cooling Interface and register each responding device as a cooling device. Later, when passive cooling is in process and the thermal limit for a zone has changed, ACPI will call this interface on each cooling device in the zone. The cooling device will then map the provided thermal limit to its specific cooling characteristics, and implement appropriate cooling mitigations. Note that if the cooling device appears in multiple thermal zones, the thermal limit that constrains the device the most (that is, the lowest percentage) is passed to the device.

Note Windows provides built-in implementations of thermal throttling for processors, backlight, and ACPI control method battery.

Thermal cooling interface

Windows provides the extension points for device drivers to register as thermal throttling devices and to receive the thermal throttling percentage request. The device is then responsible for translating that percentage to an action that makes sense for itself.

Devices that want to be added as a thermal throttling device should first add the _HIDs into the thermal zone thermal device list and then implement the following set of interfaces. As each device driver is loaded, ACPI will query for this interface and register each responding device as a cooling device. Later, when passive cooling is in process and the thermal limit for a zone has changed, ACPI will call this interface on each cooling device in the zone. The cooling device will then map the provided thermal limit to its specific cooling characteristics, and implement appropriate cooling mitigations. Note that if the cooling device appears in multiple thermal zones, the thermal limit that constrains the device the most (that is, the lowest percentage) is passed to the device.

```
//
// Thermal client interface (devices implementing
// GUID_THERMAL_COOLING_INTERFACE)
//

typedef
_Function_class_(DEVICE_ACTIVE_COOLING)
VOID
DEVICE_ACTIVE_COOLING (
    _Inout_opt_ PVOID Context,
    _In_ BOOLEAN Engaged
    );
```

```
typedef DEVICE_ACTIVE_COOLING *PDEVICE_ACTIVE_COOLING;

typedef
_Function_class_(DEVICE_PASSIVE_COOLING)
VOID
DEVICE_PASSIVE_COOLING (
    _Inout_opt_ PVOID Context,
    _In_ ULONG Percentage
    );

typedef DEVICE_PASSIVE_COOLING *PDEVICE_PASSIVE_COOLING;

typedef struct _THERMAL_COOLING_INTERFACE {
    //
    // generic interface header
    //
    USHORT Size;
    USHORT Version;
    PVOID Context;
    PINTERFACE_REFERENCE    InterfaceReference;
    PINTERFACE_DEREFERENCE  InterfaceDereference;
    //
    // Thermal cooling interface
    //
    ULONG Flags;
    PDEVICE_ACTIVE_COOLING ActiveCooling;
    PDEVICE_PASSIVE_COOLING PassiveCooling;
} THERMAL_COOLING_INTERFACE, *PTHERMAL_COOLING_INTERFACE;

#define THERMAL_COOLING_INTERFACE_VERSION 1
```

Processor

For a processor, the thermal manager communicates the thermal throttling percentage to the processor power manager (PPM). Thermal throttling of processors is a built-in feature of Windows.

Processor aggregator

The processor aggregator device allows *core parking* as a thermal mitigation. Zones can specify core parking as a thermal mitigation if the processor aggregator device is a member of a thermal zone. Processors are not required to be throttled for core parking to occur. This implementation works in parallel with *logical processor idling* (LPI). Note that this is still subject to the caveats of core parking. In particular, any work affinitized to a parked core will cause the core to run.

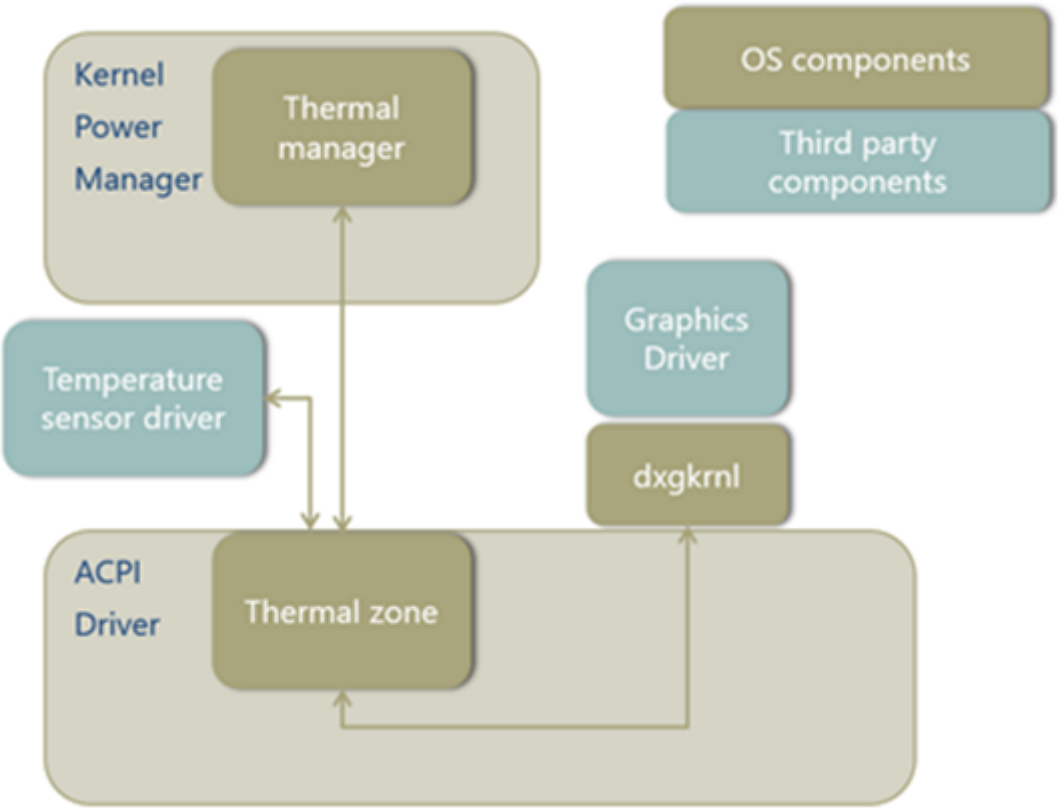
```

Device(\_SB.PAGR) {
    Name(_HID, "ACPI000C")
```

```
}
ThermalZone(\_TZ.TZ01) {
    Name(_TZD, Package() {"\_SB.PAGR"})
    // Other thermal methods: _PSV, etc.
}
```

Graphics

In order for a third-party graphics miniport driver to be thermally managed, it must interact with the Windows graphics port driver, Dxgkrnl.sys. Dxgkrnl.sys has the thermal cooling interface and passes any throttle requests down the stack to the miniport driver. It is the responsibility of the miniport driver to translate the request to an action specific to its device.

The following block diagram shows the architecture of the thermal zone that controls the GPU.



Backlight

Reducing the backlight can dramatically improve thermal conditions on a mobile platform. Windows recommends that while operating, the system display brightness is never thermally limited to less than 100 nits.

For backlight control, the thermal manager communicates the thermal throttling percentage to the monitor driver, Monitor.sys. Monitor.sys will decide the actual backlight level setting based on this thermal input and other inputs in Windows. Then

Monitor.sys will apply the backlight level setting through either ACPI or the display driver.

Note that if the thermal zone temperature that includes the backlight continues rising, the requested thermal throttling percentage might eventually drop to zero percent. The backlight level implementation in ACPI or display driver should ensure that the minimal brightness level available for performance controls is still visible for end users.

Note While operating, the system display brightness is never thermally limited to less than 100 nits.

Battery

Another main heat source in the system is battery charging. From a user's perspective, charging should be reduced and even completely halted under constrained thermal conditions. However, battery charging should not be hindered under normal use cases.

Note We recommend that battery charging is not throttled while the system is (1) idle and within the ambient temperature range below 35°C, or (2) under any conditions while the ambient temperature is below 25°C.

The Windows Control Method Battery miniclass driver, Cmbatt.sys, uses the Thermal Cooling Interface directly, as described above. Firmware is responsible for taking the current thermal limit into account when engaging charging. A new ACPI control method is necessary to set the thermal limit for charging. This method is implemented as a Device Specific Method (_DSM), as defined in the ACPI 5.0 specification, Section 9.14.1.

To apply the thermal throttling percentage, Cmbatt.sys will evaluate the Device Specific Method (_DSM) control method to request the ACPI firmware to set the thermal limit for charging. Within the _DSM control method, a GUID is defined to indicate the thermal limit.

Arg #	Value	Description
Arg0	4c2067e3-887d-475c-9720-4af1d3ed602e	UUID
Arg1	0	Revision
Arg2	1	Function
Arg3	Integer value from 0 to 100	Thermal limit for battery charging. Equivalent to calculated percentage passive throttle.

Arg #	Value	Description
Return value	N/A	No return value

Active cooling

From the point of view of the operating system, a platform has two strategies that it can use to implement fan control:

- **Implement one or more ACPI thermal zones with active trip points to engage/disengage the fan.**

  The Windows thermal framework supports active cooling devices at a very basic level. The only device supported inbox is an ACPI fan, and it can only be controlled with on/off signal.

- **Implement a proprietary solution to control the fan (via drivers, an embedded controller, etc.).**

  While Windows does not control the behavior of proprietary solutions for fans, Windows does support fan notifications to thermal manager for all implementations including embedded controllers so that diagnostic information and telemetry can be collected. Thus fan exposure to the operating system is required for all Modern Standby PCs and strongly recommended for all others.

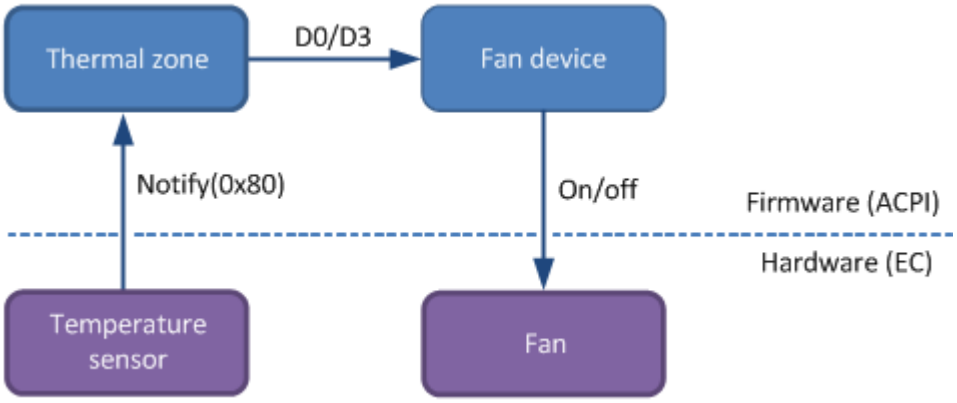
Note that the implementation for active cooling is completely separate from the passive cooling mitigations previously discussed.

Fan control by ACPI thermal zone

Windows supports the ACPI 1.0 D-state-based fan definition. (For more information, see the ACPI specification.) Thus, the control is limited to fan *on* and *off*. The driver for the fan is provided in the Windows ACPI driver, Acpi.sys.

1. The temperature sensor reads the temperature has crossed a trip point and issues Notify(0x80) on the associated thermal zone.
2. The thermal zone reads the temperature with the _TMP control method and compares the temperature to the active trip points (_ACx) to decide whether the fan needs to be on or off.
3. The operating system puts the fan device in D0 or D3 which causes the fan to turn on or off.

The following block diagram shows the control flow for a fan controlled by an ACPI thermal zone.




```
Scope(\_SB) {
    Device(FAN) {
        Name(_HID, EISAID("PNP0C0B"))
        // additional methods to turn the fan on/off (_PR0, etc.)
    }

    ThermalZone(TZ0) {
        Method(_TMP) {...}
        Name(_AC0, 3200)
        Name(_AL0, Package() {\_SB.FAN})
    }
}
```

Multiple-speed fan in ACPI

To achieve multiple speeds for a fan using ACPI 1.0, there are two options:

- The thermal zone can contain multiple "fans", when only one physical fan exists. Having more "fans" on at one time translates to faster fan speed. For more information, see the example of this option in Section 11.7.2 in the ACPI 5.0 specification.
- When the fan is on, it can decide for itself how fast to spin. Systems with embedded controllers, for example, can choose this option.

Proprietary solution for fans

Windows needs to be able to detect fan activity with either implementation. When a platform uses the ACPI thermal model, Windows is responsible for turning the fan on and off and therefore already knows when it is active. When a proprietary solution is used to control the fan, Windows needs notification that the fan is running. To enable

this, Windows will support a partial subset of the ACPI 4.0 fan extensions, which are listed in the following table.

Feature	Description	Supported
_FST	Returns the fan's status.	yes
Notify(0x80)	Indicates the fan's status has changed.	yes
_FIF	Returns fan device information.	no
_FPS	Returns a list of fan performance states.	no
_FSL	Sets the fan performance state (speed).	no

Windows will use the _FST object to determine if the fan is running (Control field is nonzero) or off (Control field is zero). Windows will also support Notify(0x80) on the fan device as an indication that _FST has changed and needs to be reevaluated.

A fan that implements the _FST object is not required to be in a thermal zone's _ALx device list, but it can, as an option, be in this list. This option enables a hybrid solution in which a fan is typically controlled by a third-party driver, but can be controlled by the OS thermal zone if the third-party driver is not installed. If a fan is in an _ALx device list and is engaged by the thermal zone (placed in D0), the _FST object is required to indicate a nonzero Control value.

For all fans, Windows will use the following algorithm determine the state of the fan:

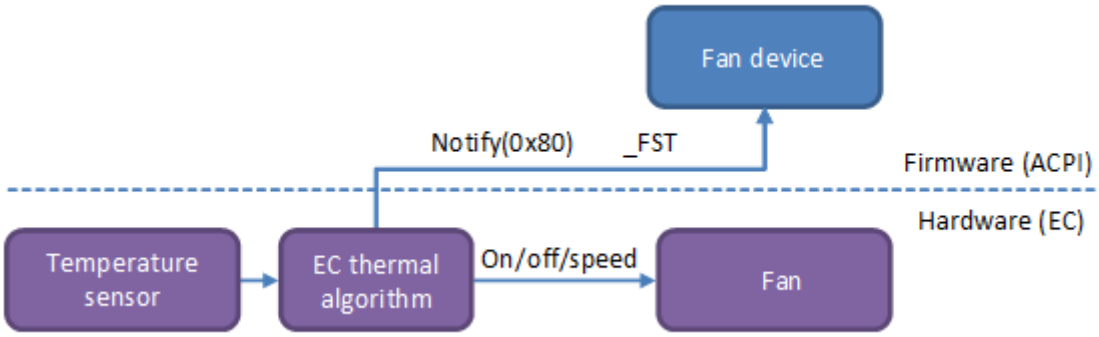
1. If a fan is in D0 (as a result of a thermal zone's _ACx trip point being crossed), it is engaged.
2. If a fan is in D3 and does not support the ACPI 4.0 extensions, it is disengaged.
3. If a fan is in D3 and supports the ACPI 4.0 extensions, the operating system will check _FST's Control field for a nonzero value to see if the fan is engaged.

Example 1: Hardware fan control

In this example, a fan is controlled by an embedded controller.

1. The embedded controller decides to start or stop the fan based on its own internal algorithm.
2. After starting or stopping the fan, the embedded controller issues a Notify(0x80) on the fan device.
3. The operating system evaluates _FST and reads the new state of the fan.

The following block diagram shows the control flow for a fan controlled by an embedded controller.



The following ASL example defines a "FAN" device that can notify the operating system of changes in the fan state:

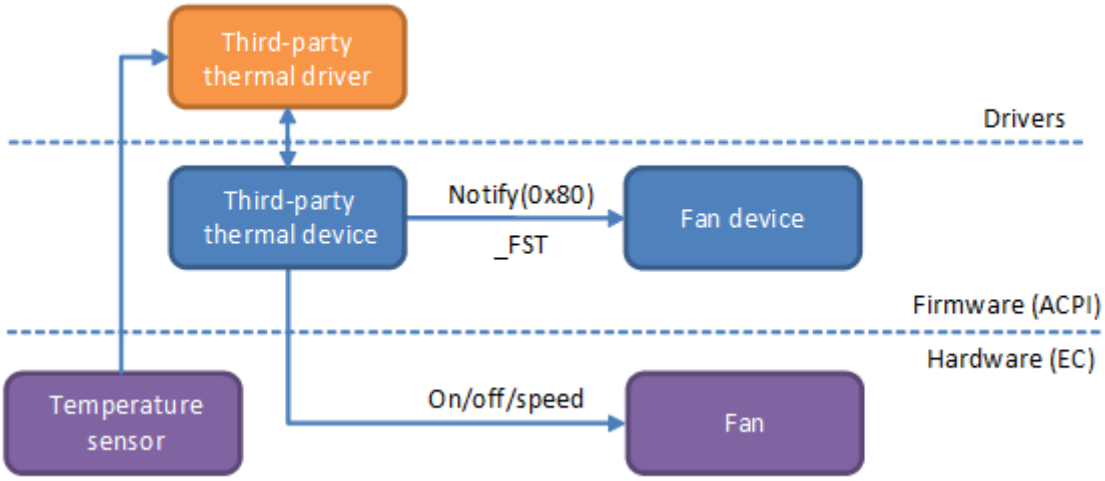
```
Scope(\_SB) {
    Device(FAN) {
        Name(_HID, EISAID("PNP0C0B"))
        Name(_FST, Package() {0, 0, 0xffffffff})

        // \_SB.FAN.SFST called by EC event handler upon fan status change
        Method(SFST, 1) {
            // Arg0 contains the new fan status
            Store(Arg0, Index(_FST, 1))
            Notify(\_SB.FAN, 0x80)
        }
    }

    // Omitted: EC event handler
}
```

Example 2: Driver fan control

In this example, a third-party driver is controlling the fan.

1. The driver decides to start or stop the fan based on its own internal algorithm.
2. The driver modifies the state of the fan and evaluates a custom control method (SFST) on its thermal device.
3. The thermal device control method updates the fan device's _FST contents and issues Notify(0x80) on the fan device.
4. The operating system evaluates _FST and reads the new state of the fan.

The following block diagram shows the control flow for a fan controlled by a third-party driver.




```
Scope(\_SB) {
    Device(FAN) {
        Name(_HID, EISAID("PNP0C0B"))
        Name(_FST, Package() {0, 0, 0xffffffff})
    }

    Device(THML) {
        Name(_HID, "FBKM0001")
        Method(SFST, 1) {
            // Arg0 contains the new fan status
            Store(Arg0, Index(\_SB.FAN._FST, 1))
            Notify(\_SB.FAN, 0x80)
        }
    }
}
```

Fan presence

A platform indicates that there is a fan on the system by including a fan device (PnP ID PNP0C0B) in the ACPI namespace. Windows will take the presence of this device as an indication that the system has a fan, and the absence of this device as an indication that the system has no fan.

Guidance specific to Modern Standby

The Windows thermal management framework is a part of the kernel and ships with all Windows systems. Thus, the material above applies to all machines. However, various types of machines require additional guidance more specific to Modern Standby.

Modern Standby brings the smart phone power model to the PC. It provides an instant on, instant off user experience that users have come to expect on their phone. And just

like on the phone, Modern Standby enables the system to stay fresh, up-to-date, and reachable whenever a suitable network is available. Windows 10 supports Modern Standby on low-power platforms that meet specific Windows certification requirements.

Modern Standby PCs are highly mobile devices in a thin and light form factor. Further, Modern Standby PCs are always on and in the ACPI S0 state. In order to deliver a robust and reliable customer experience, the entire system—from the mechanical design to firmware and software implementation—must be designed with critical attention to thermal characteristics. Thus, all Modern Standby PCs must adhere to thermal requirements.

Modern Standby requirements

All Modern Standby PCs must pass the following HCK tests:

- All Modern Standby PCs must have at least one thermal zone for which a critical shutdown temperature (_CRT) is defined. For x86 systems, we recommend defining a critical hibernate temperature (_HOT) to trigger the saving of user data. _HOT must be lower than _CRT, and _CRT should be lower than the firmware fail-safe thermal trip point.
- Each thermal zone must report actual temperature on the sensor (_TMP). The test runs a varying workload on the PC, and the temperature is expected to change.

In addition, we recommend that each PC includes at least one temperature sensor for the SoC.

Active cooling requirements

Modern Standby PCs that use fans must adhere to the following additional requirements, which are tested in HCK:

- The fan must be exposed to the operating system. For more information, see Fan presence.
- The operating system must know at all times whether the fan is on or off. Even while in idle resiliency in Modern Standby, any changes in fan activity must wake the SoC to update the status. For more information about implementing fan notifications, see Fan control by ACPI thermal zone.
- **The fan must not turn on while the PC is in Modern Standby.** With a realistic workload during Modern Standby, which is whenever the screen is off, the fan must not turn on.

From a user perspective, the PC appears to be turned off when it is in Modern Standby. Users expect a PC in Modern Standby to behave as though it is in the system "sleep" state. Thus, users expect the fan to never come on, just as in traditional PCs during

sleep. If the fan does come on, users might hear it and/or feel the hot air circulating and think that the computer is not working correctly. Therefore, the fan should not turn on while in Modern Standby. For more information about the required behavior, see HCK test requirements for modern standby PCs.

These requirements imply that the cooling policy when the screen is turned on might need to be different from when the screen turned off. The PC might use active cooling while the screen is on, but it must rely only on passive cooling when the screen is off. The active trip point for the fan might be much lower when the screen is on than when it's off. For the ACPI implementation, _ACx would have to be updated on entry to Modern Standby. For proprietary solutions, please make sure to update the policy in the embedded controller.

Processor throttling

The PPM communicates the maximum, desired and minimal performance levels to the PEP. Under thermal throttling conditions, the maximum performance level should equal the throttling performance requested by the thermal manager. The PEP then sets the CPU's physical voltage and frequency based on the PPM's performance level requirements.

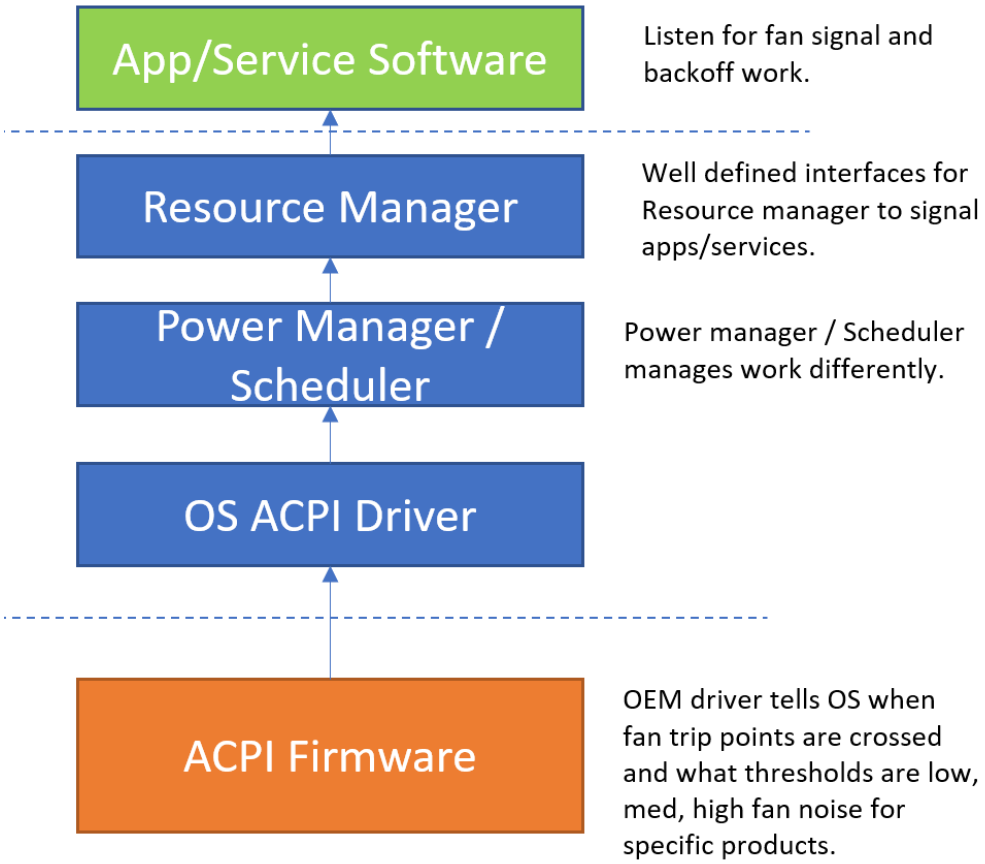
Fan Noise Signal

Starting in Windows 11, devices can send a fan noise signal to the OS for use in resource management decisions, with the goal of providing users with a cool and quiet experience. This opt-in interface allows firmware to send fan RPM information to the OS as a value from 0 (fan off) to Max RPM, which the OS can use in resource management decisions to cool the device as needed. This contributes to two major benefits:

1. **Quiet User Experience:** Fan noise and hot air blowing are a significant source of customer dissatisfaction. This is especially true when the user is not expecting heavy fan activity, such as when the device is running a low amount of work or no foreground work.
2. **Improved Battery Life:** Higher fan speeds result in more power usage, which causes faster battery drain while on DC and slower charging while on AC.

Currently, this signal can be used to manage Search Indexer tasks, and there are plans to enable additional background tasks to consume this signal as well.

The following diagram summarizes how fan noise signal is sent in the layers from firmware to software.

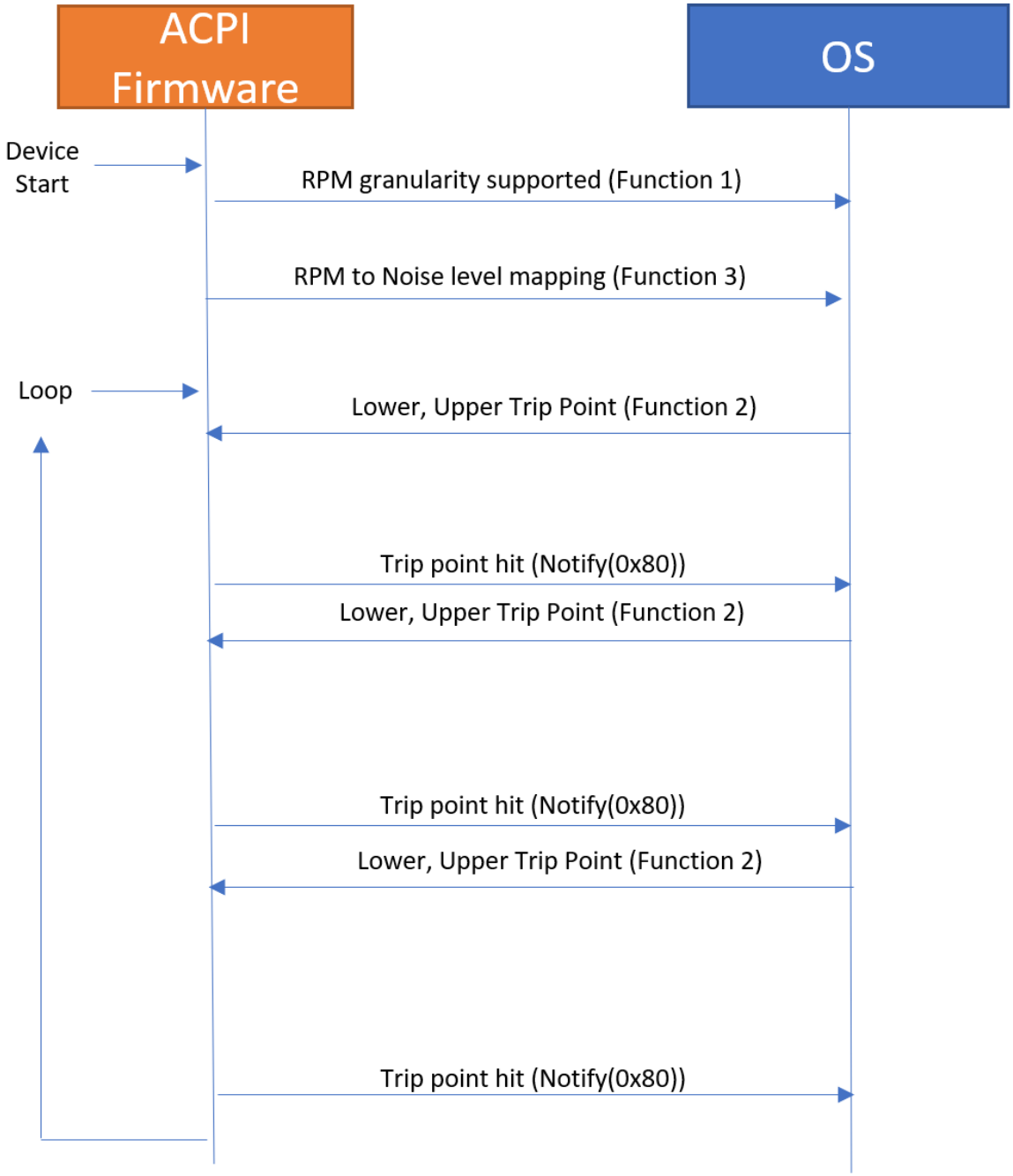

App/Service Software
Listen for fan signal and backoff work.
Resource Manager
Well defined interfaces for Resource manager to signal apps/services.
Power Manager / Scheduler
Power manager / Scheduler manages work differently.
OS ACPI Driver
ACPI Firmware
OEM driver tells OS when fan trip points are crossed and what thresholds are low, med, high fan noise for specific products.

How Fan Noise Signal Works

ACPI Interface Details

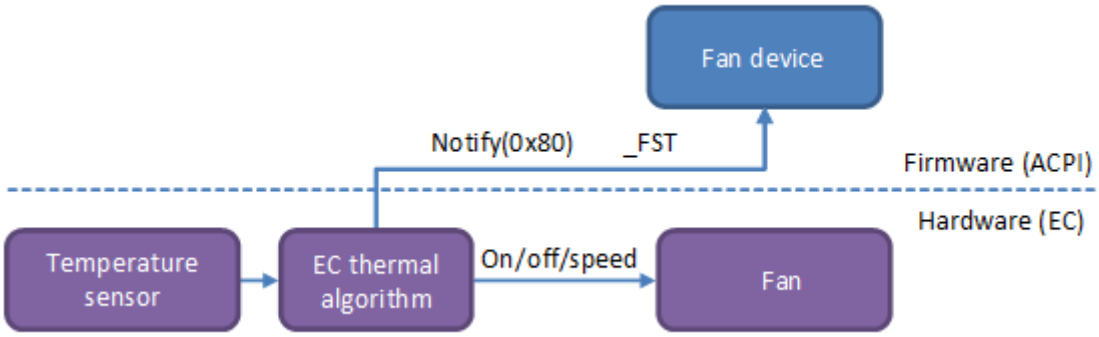
Below are the four functions of Device Specific Method (_DSM, UUID: A7611840-99FE-41ae-A488-35C75926C8EB) used to support this feature. Information on _FST (Fan Status) can be found on section 11.3.1.4 of the ACPI specification and in **Example 1: Hardware fan control** of Thermally Managed Devices

The following diagram summarizes the flow of how these functions are used.


ACPI Firmware
OS
Device Start
RPM granularity supported (Function 1)
RPM to Noise level mapping (Function 3)
Loop
Lower, Upper Trip Point (Function 2)
Trip point hit (Notify(0x80))
Lower, Upper Trip Point (Function 2)
Trip point hit (Notify(0x80))
Lower, Upper Trip Point (Function 2)
Trip point hit (Notify(0x80))

The following block diagram shows the control flow for a fan controlled by an embedded controller, including the Notify(0x80) control method.

> ⓘ **Note**
>
> Fan Noise Signal would not control Fan RPM but instead notify the OS there is fan noise that requires moving CPU resource from background tasks to complete priority work.



Enumerate Functions (Function 0)

For the operating system to interact with the platform, an ACPI device must be exposed through the Namespace. This Device must include a _CID object containing EISAID("**PNP0C0B**"). This device's scope must contain the following _DSM definition indicating which _DSMs the device supports.

UUID	Revision	Function	Description
A7611840-99FE-41ae-A488-35C75926C8EB	0	0	Enumerate functions

Return: In order to indicate support for functions 0 through 3 listed above, the Enumerate Functions function (function 0) should return 0xF. Please refer to section 9.1.1 of the ACPI spec for more information.

Get Fan Trip Point Support Function (Function 1)

Hardware tells OS what's supported in terms of RPMs.

UUID	Revision	Function	Description
A7611840-99FE-41ae-A488-35C75926C8EB	0	1	Get fan trip point support

Arguments

Arg0: UUID: A7611840-99FE-41ae-A488- 35C75926C8EB

Arg1: Revision: 0

Arg2: Function: 1

Arg3: An empty Package

Return: An integer value containing the granularity supported for fan trip points, in RPMs. If non-zero, OS may select trip points which are a multiple of the notification granularity. For example, with a granularity of 200, OSPM would be allowed to select trip points at 0, 200, 400, 600, etc. RPMs. A value of 0 indicates trip points are not supported.

Set Fan Trip Points Function (Function 2)

OS communicates to hardware what the next notification trip point is; hardware notifies OS when that happens.

UUID	Revision	Function	Description
A7611840-99FE-41ae-A488-35C75926C8EB	0	2	Get fan trip points

Arguments

Arg0: UUID: A7611840-99FE-41ae-A488-35C75926C8EB

Arg1: Revision: 0

Arg2: Function: 2

Arg3: A Package containing the lower and upper trip points. (2 element int. Lower at index 0, Upper at index 1)

OSPM will only select trip points which are a multiple of the trip point granularity specified in function 1. When the actual speed of the fan goes above or below the upper/lower trip points, the platform should issue Notify(0x80) on the fan device. OSPM will then evaluate _FST (Fan status) to determine the current fan speed. If the fan speed is already outside of the specified trip points when the trip points are set, the platform should immediately issue Notify(0x80).

The upper trip point will be a multiple of granularity. The lower trip point will be (multiple of granularity) + 1 (lower trip point < upper trip point). When RPM is 0, OS will set lower trip point to 0, and upper trip point to 1.

Return: None.

Get Fan Operating Ranges Function (Function 3)

Mapping between RPM –> Impact. Please note that only one fan (the one closest to the SoC) can implement this interface, it must implement all 3 functions plus function 0 from section 9.1.1 of the ACPI spec.

UUID	Revision	Function	Description
A7611840-99FE-41ae-A488-35C75926C8EB	0	3	Get fan operating ranges

Arguments

Arg0: UUID: A7611840-99FE-41ae-A488-35C75926C8EB

Arg1: Revision: 0

Arg2: Function: 3

Arg3: An empty Package.

Return: A Package containing the following format:

```
Package () {
    Impact1MaxRPM,      // Integer (DWORD)
    Impact2MaxRPM,      // Integer (DWORD)
    Impact3MaxRPM,      // Integer (DWORD)
    MaxRPM              // Integer (DWORD)
}
```

Field	Format	Description
Impact1MaxRPM	Integer (DWORD)	Maximum RPMs for fan impact range 1.
Impact2MaxRPM	Integer (DWORD)	Maximum RPMs for fan impact range 2. Must be >= Impact1MaxRPM.
Impact3MaxRPM	Integer (DWORD)	Maximum RPMs for fan impact range 3. Must be >= Impact2MaxRPM.
MaxRPM	Integer (DWORD)	Maximum RPMs the fan can operate at. Must be >= Impact3MaxRPM.

This table is used to derive the RPM range for each of the fan impact levels:

Impact Score	Lower RPM Value	Upper RPM Value
1	1	Impact1MaxRPM
2	Impact1MaxRPM + 1	Impact2MaxRPM.

Impact Score	Lower RPM Value	Upper RPM Value
3	Impact2MaxRPM + 1	Impact3MaxRPM
4	Impact3MaxRPM + 1	MaxRPM

If an impact range is not used by a platform (for example, if the fan transitions directly from impact range 2 to impact range 4), it can indicate this by setting the Max RPM for the unused impact range equal to the max RPM for the lower impact range.

Example Mapping

Value Sent to OS	Fan RPM	User Experience	RPM Ceiling
0 – Low	1-4000 RPM (<=25 dBA)	Fan not on or on with low annoyance	Impact1MaxRPM = 4000
1 – Medium	4001-5000 RPM (25-30 dBA)	Fan on with medium annoyance	Impact2MaxRPM = 5000
2 – Medium-High	5001-6000 RPM (30-36 dBA)	Fan on with medium-high annoyance	Impact3MaxRPM = 6000
3 – High	6001+ RPM (36+ dBA)	Fan on with high annoyance	MaxRPM = 9000

Example ASL Code

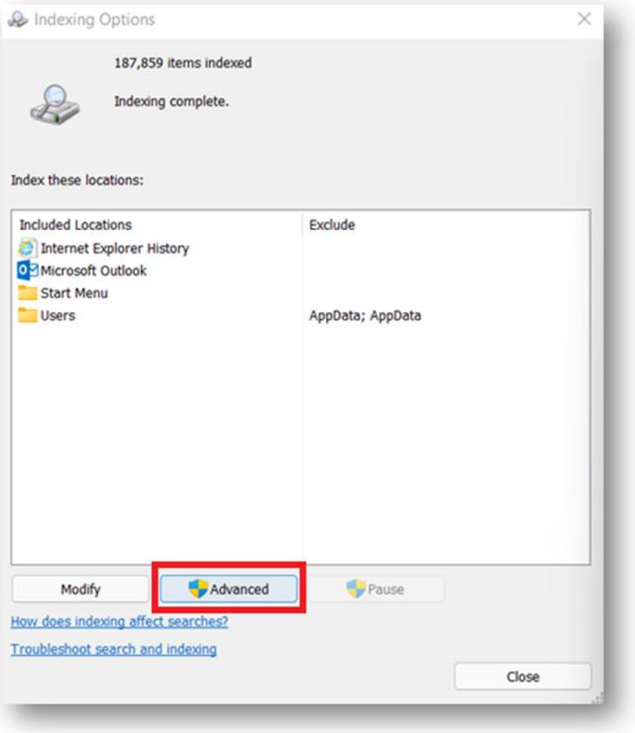
```
    ...

    // _DSM - Device Specific Method
    // Arg0: UUID Unique function identifier
    // Arg1: Integer Revision Level
    // Arg2: Integer Function Index (0 = Return Supported Functions)
    // Arg3: Package Parameters
    Method(_DSM, 0x4, NotSerialized) {
            If(LEqual(Arg0, ToUUID("A7611840-99FE-41ae-A488-35C75926C8EB")))
{
                Switch (ToInteger(Arg2)) {
                    Case(0) {
                        // _DSM functions 0 through 3 are supported
                        Return (Buffer() {0xf})
                    }
                    Case(1) {
                        // Report 200 RPM granularity for trip points
                        Return (\_SB.FAN0.GRAN)
                    }
```

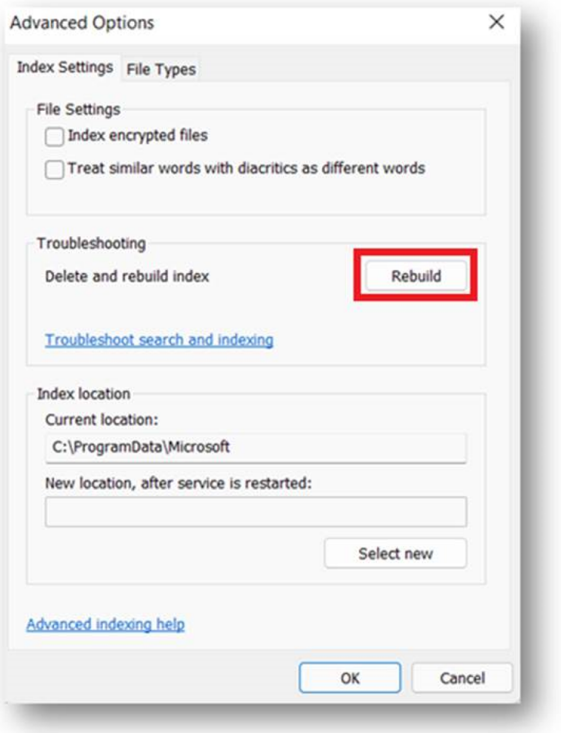
```
                    Case(2) {
                        // Save lower RPM trip point
                        Store(DeRefOf(Index(Arg3, 0)), \_SB.FAN0.LRPM)
                        // Save upper RPM trip point
                        Store(DeRefOf(Index(Arg3, 1)), \_SB.FAN0.URPM)
                        // Configure hardware for the trip points, Tell EC
to set fan speed trip point.
                        \_SB.FAN0.CRNF()
                        Return (0)
                    }
                    Case(3) {
                        Return (Package(4) {
                            4000, // 1-4000 RPM is impact score 1
                            5000, // 4001-5000 RPM is impact score 2
                            6000, // 5001-6000 RPM is impact score 3
                            9000})// 6001-9000 RPM is impact score 4
                    }
                    Default {
                        Return(Buffer(One) { 0x00 }) // Guid mismatch
                    }
                }
            }
            Else {
                Return(Buffer(One) { 0x00 }) // Guid mismatch
            }
    }

} // end of FAN0
}
```

Testing and Tracing

Please reference following steps to collect log and view in Windows Performance Analyzer (WPA):

1. Settings -> Searching Windows -> Advanced Search Indexer Settings -> Advanced -> Delete and rebuild index (Rebuild)
2. wpr -boottrace -addboot AcpiFanNoiseImpact.wprp –filemode
3. Restart the system
4. Check Index status of Settings -> Searching Windows (make sure device connect with AC-power source.)
5. When index is completed, stop the trace: wpr -boottrace -stopboot AcpiFanNoiseImpact.etl (Cancel trace without saving: wpr -boottrace –cancel)
6. Open AcpiFanNoiseImpact.etl via Windows Performance Analyzer (WPA) .





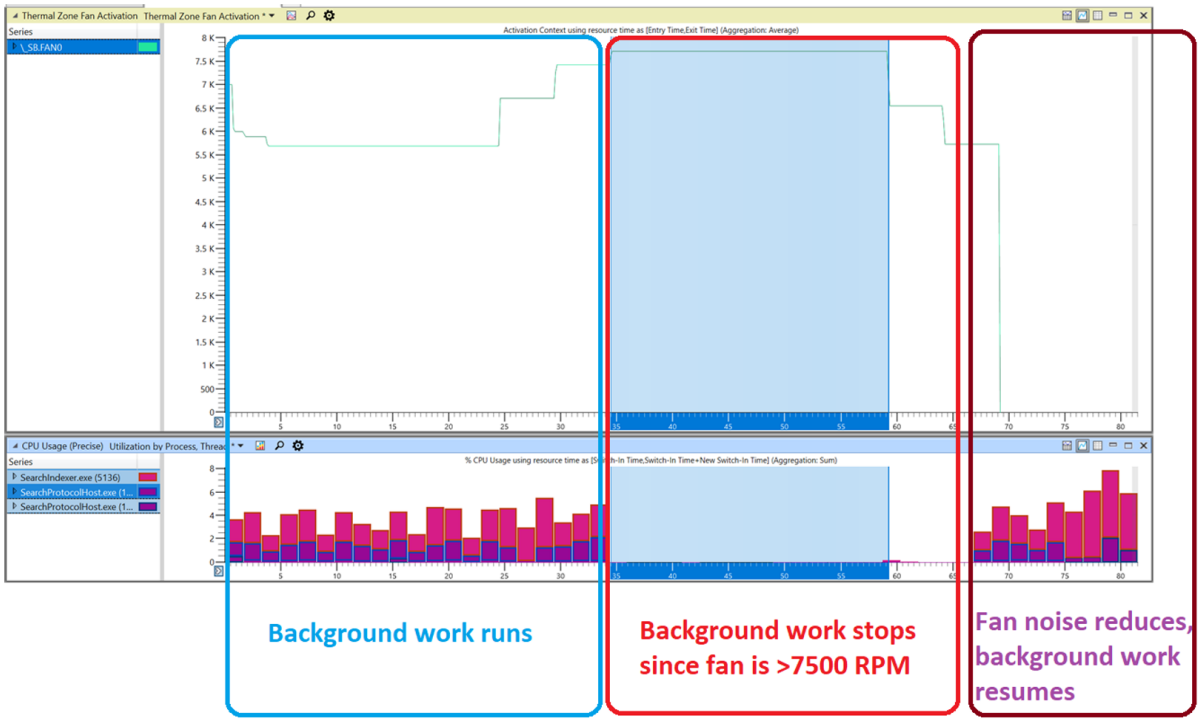
Download the file at AcpiFanNoiseImpact.zip Or, copy the following and save as **AcpiFanNoiseImpact.wprp**

```
<?xml version="1.0" encoding="utf-8"?>
<WindowsPerformanceRecorder Version="1.0" Comments="Test" Company="Microsoft
Corporation" Copyright="Microsoft Corporation">
    <Profiles>
        <!-- BufferSizes are in KB in WPRP -->
        <!-- System Collectors -->
        <SystemCollector Id="MySystemCollector" Name="NT Kernel Logger">
            <BufferSize Value="1024" />
            <Buffers Value="100" />
            <StackCaching BucketCount="2048" CacheSize="20480" />
            <FlushThreshold Value="70" />
        </SystemCollector>

        <!-- Event Collectors -->
        <EventCollector Id="MyEventCollector" Name="User Session Logger">
            <BufferSize Value="1024" />
            <Buffers Value="100" />
            <StackCaching BucketCount="2048" CacheSize="20480" />
            <FlushThreshold Value="70" />
        </EventCollector>

        <!-- System Providers for collecting kernel events. -->
        <SystemProvider Id="SP_AcpiFanNoiseImpactTrace">
            <Keywords Operation="Add">
                <Keyword Value="Loader" />
                <Keyword Value="Power" />
                <Keyword Value="ProcessThread" />
```

```
            </Keywords>
        </SystemProvider>
        <!-- System Providers for collecting kernel events. -->
        <!---->

        <EventProvider Id="EP_Microsoft-Windows-Kernel-Power"
Name="Microsoft-Windows-Kernel-Power" Level="5" NonPagedMemory="true">
            <Keywords>
                <Keyword Value="0x2" />
            </Keywords>
            <CaptureStateOnStart>
                <Keyword Value="0x0" />
            </CaptureStateOnStart>
            <CaptureStateOnSave>
                <Keyword Value="0x0" />
            </CaptureStateOnSave>
        </EventProvider>
        <EventProvider Id="EP_Microsoft-Windows-Kernel-Acpi"
Name="Microsoft-Windows-Kernel-Acpi" Level="5">
            <Keywords>
                <Keyword Value="0xffffffff" />
            </Keywords>
            <CaptureStateOnSave>
                <Keyword Value="0xffffffff" />
            </CaptureStateOnSave>
        </EventProvider>
        <EventProvider
Id="CustomEventProvider_Microsoft.Windows.SRUM.Telemetry_TraceLogging"
Name="7073707A-0587-4E03-B31F-6443EB1ACBCD" Level="5" />
        <EventProvider
Id="CustomEventProvider_Microsoft.Windows.Kernel.Acpi_TraceLogging"
Name="C42BBFDB-4140-4ada-81DF-2B9A18AC6A7B" Level="5" />
        <EventProvider
Id="CustomEventProvider_Microsoft.Windows.Kernel.Power_TraceLogging"
Name="63bca7a1-77ec-4ea7-95d0-98d3f0c0ebf7" Level="5" />
        <EventProvider Id="CustomEventProvider_AcpiTraceGuid_WPP"
Name="03906A40-CCE8-447F-83F4-E2346215DB84" Level="7" />
        <EventProvider
Id="CustomEventProvider_Microsoft.Windows.CentralResourceManager_TraceLoggin
g" Name="8215e965-d26e-548e-af0e-940c1f06f250" Level="5"
NonPagedMemory="true">
            <CaptureStateOnSave>
                <Keyword Value="0xFFFFFFFFFFFFFFFF" />
            </CaptureStateOnSave>
        </EventProvider>

        <Profile Id="PowerTrace.Verbose.File" LoggingMode="File"
Name="PowerTrace" DetailLevel="Verbose" Description="Power trace logging">
            <Collectors>
                <SystemCollectorId Value="MySystemCollector">
                    <SystemProviderId Value="SP_AcpiFanNoiseImpactTrace" />
                </SystemCollectorId>
                <EventCollectorId Value="MyEventCollector">
                    <EventProviders>
                        <EventProviderId Value="EP_Microsoft-Windows-Kernel-
```

```
Power" />
                        <EventProviderId Value="EP_Microsoft-Windows-Kernel-
Acpi" />
                        <EventProviderId
Value="CustomEventProvider_Microsoft.Windows.Kernel.Acpi_TraceLogging" />
                        <EventProviderId
Value="CustomEventProvider_AcpiTraceGuid_WPP" />
                        <EventProviderId
Value="CustomEventProvider_Microsoft.Windows.Kernel.Power_TraceLogging" />
                        <EventProviderId
Value="CustomEventProvider_Microsoft.Windows.SRUM.Telemetry_TraceLogging" />
                        <EventProviderId
Value="CustomEventProvider_Microsoft.Windows.CentralResourceManager_TraceLog
ging" />
                    </EventProviders>
                </EventCollectorId>
            </Collectors>
        </Profile>
    </Profiles>
    <TraceMergeProperties>
        <TraceMergeProperty Id="TraceMerge_Default"
Name="TraceMerge_Default" Base="">
            <DeletePreMergedTraceFiles Value="true" />
            <FileCompression Value="true" />
            <CustomEvents>
                <CustomEvent Value="ImageId" />
                <CustomEvent Value="BuildInfo" />
                <CustomEvent Value="VolumeMapping" />
                <CustomEvent Value="EventMetadata" />
                <CustomEvent Value="PerfTrackMetadata" />
                <CustomEvent Value="WinSAT" />
                <CustomEvent Value="NetworkInterface" />
            </CustomEvents>
        </TraceMergeProperty>
    </TraceMergeProperties>
</WindowsPerformanceRecorder>
```

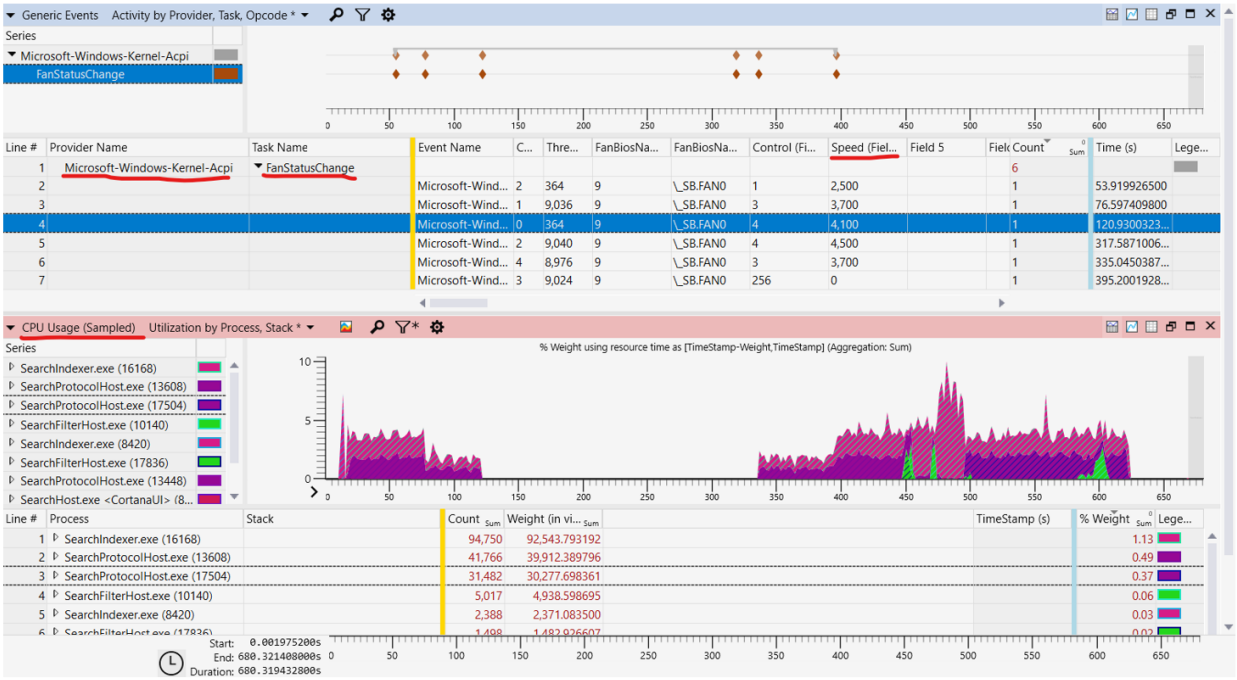
The image below is an example WPA graph showing Search Indexer work backing off when the fan is loud.



There is one level of the fan that would make search index back off in full (the highest, that should be impact score 4), but all other levels of the fan should be reduced activity and not suspension. For example, if ACPI declared Impact3MaxRPM = 4000 RPM in function 3 (Get fan operating ranges function), when Fan RPM > 4000 (4100RPM, 4500RPM), we would see SearchIndexer.exe and SearchProtocalHost.exe CPU Usage would be suspended.

> **Note**
>
> To see CPU Usage, you can use *wpr -start power -filemode* to collect runtime usage. Use *wpr -stop fan_noise.etl* to stop collecting log.

The image below shows an example WPA graph showing the suspension of SearchIndexer.exe and SearchProtocolHost


Generic Events
Activity by Provider, Task, Opcode *
Series
Microsoft-Windows-Kernel-Acpi
FanStatusChange
Line # | Provider Name | Task Name | Event Name | C... | Thre... | FanBiosNa... | FanBiosNa... | Control (Fi... | Speed (Fiel... | Field 5 | Field Count | Time (s) | Lege...
1 | Microsoft-Windows-Kernel-Acpi | FanStatusChange | 6
2 | Microsoft-Wind... | 2 | 364 | 9 | _SB.FAN0 | 1 | 2,500 | 1 | 53.919926500
3 | Microsoft-Wind... | 1 | 9,036 | 9 | _SB.FAN0 | 3 | 3,700 | 1 | 76.597409800
4 | Microsoft-Wind... | 0 | 364 | 9 | _SB.FAN0 | 4 | 4,100 | 1 | 120.9300323...
5 | Microsoft-Wind... | 2 | 9,040 | 9 | _SB.FAN0 | 4 | 4,500 | 1 | 317.5871006...
6 | Microsoft-Wind... | 4 | 8,976 | 9 | _SB.FAN0 | 3 | 3,700 | 1 | 335.0450387...
7 | Microsoft-Wind... | 3 | 9,024 | 9 | _SB.FAN0 | 256 | 0 | 1 | 395.2001928...
CPU Usage (Sampled)
Utilization by Process, Stack *
Series
SearchIndexer.exe (16168)
SearchProtocolHost.exe (13608)
SearchProtocolHost.exe (17504)
SearchFilterHost.exe (10140)
SearchIndexer.exe (8420)
SearchFilterHost.exe (17836)
SearchProtocolHost.exe (13448)
SearchHost.exe <CortanaUI> (8...
% Weight using resource time as [TimeStamp-Weight,TimeStamp] (Aggregation: Sum)
Line # | Process | Stack | Count | Weight (in vi... | TimeStamp (s) | % Weight | Lege...
1 | SearchIndexer.exe (16168) | 94,750 | 92,543.793192 | 1.13
2 | SearchProtocolHost.exe (13608) | 41,766 | 39,912.389796 | 0.49
3 | SearchProtocolHost.exe (17504) | 31,482 | 30,277.698361 | 0.37
4 | SearchFilterHost.exe (10140) | 5,017 | 4,938.598695 | 0.06
5 | SearchIndexer.exe (8420) | 2,388 | 2,371.083500 | 0.03
Start: 0.001975200s
End: 680.321408800s
Duration: 680.319432800s

User Experience

Article • 01/18/2019

From a thermal point of view, the user experience should be such that the user can operate the PC for as long as possible without interruption. And the user should only be notified of thermal problems, when the PC can no longer operate.

This uninterrupted operation of the PC should continue, even when the PC undergoes some form of thermal mitigation.

Thermal user experience

Thermal management should be as invisible as possible to end users. Slates and other system on a chip-based (SoC-based) PCs should contain hardware and software to efficiently manage case and component temperatures and to keep the hardware platform running cool under normal use. The PC should stay turned on and fully functioning without any need for thermal mitigation for as long as possible. This goal is dependent on good hardware design.

Thermal scenarios

Thermal problems fall into these three categories:

- **The PC is turned on, but is producing too much heat to be sustainable.**

  The PC is fully functioning when it detects that it is getting warm. Thermal mitigation measures (passive cooling and active cooling) must begin to reduce the heat produced, to a sustainable rate. If thermal mitigation measures are not successful, the PC will turn itself off (see the last item in this list).

- **The PC is turned on, but is unusable due to thermal constraints.**

  This problem could manifest in a few different ways:
  - The system display is on but does not respond to the user's inputs.
  - The PC is too hot to continue running and must initiate shutdown or hibernate now.
  - The PC is much too hot and firmware needs to cut the power to the system.

- **The PC is turned off, and cannot boot due to thermal constraints.**

  When a user tries to turn on a PC in an environment that is too hot or too cold, the PC fails to boot.

Thermal mitigation experience

While the PC is turned on and being used, thermal mitigation solutions should be as transparent to the user as possible. In other words, the user ideally should never know when their PC is or is not being thermally mitigated. The overarching goal of thermal mitigation is to allow the user to use the PC for as much and as long as possible without disruption.

Hardware vendors and system integrators have a great deal of control in their design of a PC to reduce the need for thermal mitigation by designing hardware that is effective at dissipating heat. For more information about hardware design for thermal management, see Hardware thermal modeling and evaluation.

In addition, Windows provides thermal management software. For more information, see Windows thermal management framework.

Critical thermal "shutdown" experience

In extreme conditions where the ambient temperature is much higher above the targeted temperature range, heavy workloads might cause the system temperature to continue rising even with the throttling mechanisms engaged.

When the critical shutdown threshold is met, the system immediately starts the critical thermal action. By default, systems will hibernate if enabled and shutdown if hibernate is not available. This thermal shutdown path is the shortest shutdown path available and usually completes within one second. No notification is given to apps or services, so apps do not have the opportunity to autosave or close.

Furthermore, each system must have knowledge of its *hardware fail-safe temperature*, which is determined in ACPI by the hardware vendor. This value is very important because the firmware must be able to detect thermal constraints before the operating system is loaded and prevent the system from causing damage to itself and the user. If at any time while the system is on, the hardware fail-safe trip point is reached, the system must immediately cut off the power and turn off. The fail-safe trip point is usually slightly higher than the critical shutdown trip point. This way, the system has the opportunity to hibernate or shutdown before the hardware fail-safe is reached. It is possible to reach the fail-safe trip point before the system has completed a critical hibernate or shutdown. However, if the critical threshold is chosen well, this should seldom occur.

Thermal boot experience

Systems have constraints during boot as well. In firmware, each system must check the temperature before boot to ensure that boot can complete successfully. The two

primary concerns are:

- **The ambient temperature is too hot.**

  Boot is often a resource-intensive action, and, as a result, produces a lot of heat. If too much heat is produced during boot, the system might hit the hardware fail-safe temperature and kill power to itself.

- **The ambient temperature is too cold.**

  The battery might not be able supply enough resources to the system.

System designers should model and predict the thermal characteristics of boot — in particular, the temperature rise. Firmware must check that there is enough headroom for the system to boot without crossing the hardware fail-safe trip point. For example, if the system temperature consistently rises 5°C during boot, and the fail-safe temperature is 40°C, the system should be at the very most 35°C at the start of boot. The firmware must perform this check because the Windows thermal framework will not have been loaded at boot time to mitigate the issue.

Note If a PC is too hot or too cold to boot, the PC should provide user feedback to give the user an opportunity to alleviate the problem and to try to boot again.

Examples, Requirements and Diagnostics

Article • 05/18/2022

This topic presents examples of thermal management issues, and also discusses requirements and diagnostic methods.

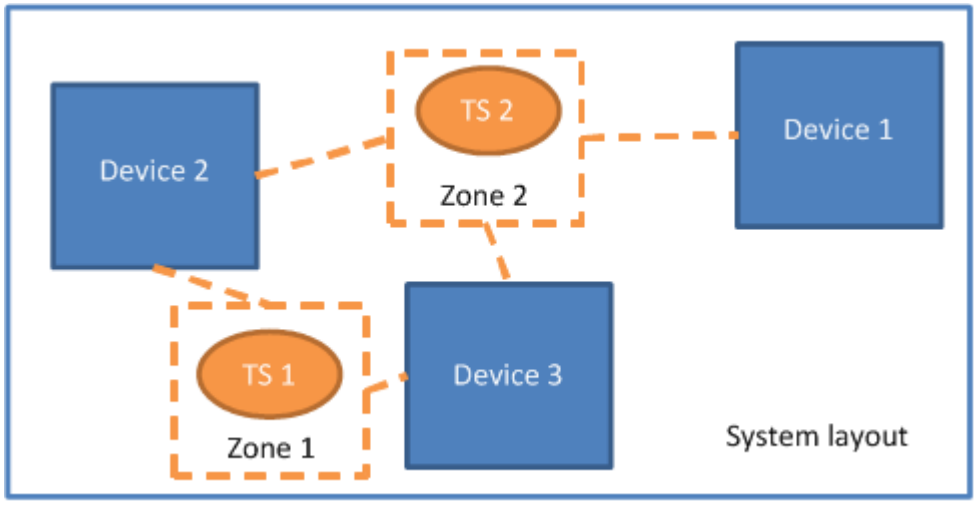
Examples

The following examples explain how to address typical thermal management issues.

Skin temperature sensors

Monitoring the skin temperature is critical in ensuring that the user is protected at all times. If the temperature on the skin is too hot to handle safely, the system should take immediate action to shut down the system. These temperature sensors can also give input to the thermal zones to throttle devices that contribute to its reading.

The following block diagram shows an example system layout that has three devices and two thermal zones.



In this example, Temperature Sensor 1 (TS1) and Temperature Sensor 2 (TS2) are placed strategically at locations where the devices contribute the most heat to the skin. Devices 1, 2, and 3 may have individual temperature sensor on top of each device. These device sensors are geared towards throttling each device individually. Usually, the purpose of the skin sensor is to detect the temperature on the surface of the device as an aggregate of multiple devices on the system. Although each device might produce more heat than can be detected at these temperature sensors, the combined heat production of these devices tends to accumulate at these sensor locations.

TS1 is placed midway between Device 2 and Device 3. Thus, the thermal zone that takes TS1 as input controls Device 2 and Device 3. When TS1 gets hot, the thermal zone throttles Device 2 and 3. Similarly, when TS2 gets hot, the thermal zone throttles all three devices.

In this example, the sensors are placed equally far from the devices that they monitor. TS1 is placed midway between Device 2 and Device 3, and TS2 is placed equidistant from Devices 1, 2, and 3. If each device dissipates heat radially the same way, the heat from each device contributes equally to the temperature reading on its sensor.

Gradual thermal throttling

Given a set of thermal constants (_TC1 and _TC2), the passive throttling percentage of a thermal zone has certain characteristics: how quickly the curve changes, and how aggressively the zone throttles to remain far from the trip point. In some circumstances, the behavior of the thermal zone might need to change. For example, when the temperature is low, the throttle percentage can afford to be less aggressive. But when the temperature is closer to the trip point, the throttling behavior might need to be much more aggressive. If so, gradual thermal throttling can be used to apply different throttling behaviors to a set of devices. There are two ways of implementing gradual thermal throttling:

- Dynamically change the constants for a thermal zone during runtime, or
- Using two thermal zones with different constants and trip points.

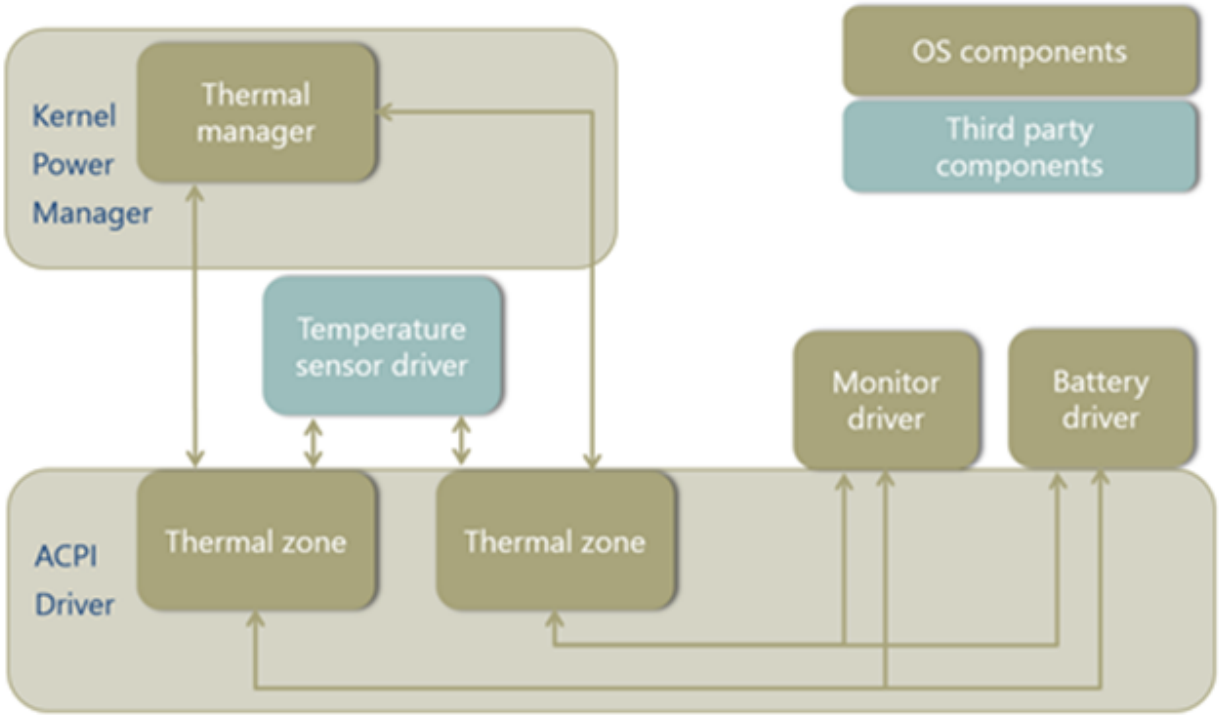
Updating constants for zones

For any thermal zone, a `Notify(thermal_zone, 0x81)` can be used to update the thermal constants at any time.

Zones with different trip points

There can be no more than one thermal sensor in a thermal zone. However, multiple thermal zones that share the same temperature sensor are frequently used to implement gradual thermal throttling behavior. One thermal zone starts to throttle performance moderately at low temperatures while the other thermal zone starts to throttle performance aggressively at high temperatures.

In the following block diagram, two thermal zones that manage the same devices use the same temperature sensor to achieve gradual thermal throttling. In this example, the temperature sensor is placed near the battery charger and the monitor backlights so that it can provide inputs to thermal zones that control these two devices.



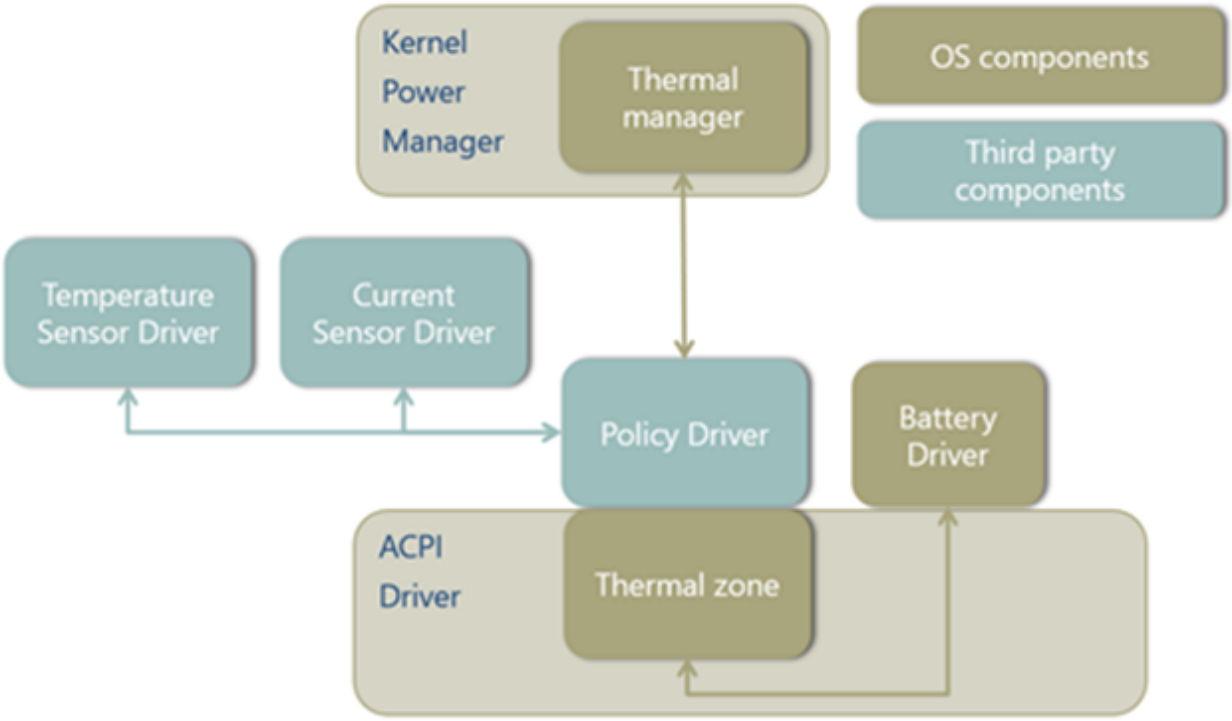
The two thermal zones shown in the preceding diagram might be defined as follows:

```
Thermal Zone 1 {     _PSV = 80C  Thermal Throttling  Devices:    Monitor
Driver  Battery Driver } Thermal Zone 2 {   _PSV = 90C  Thermal Throttling
Devices:    Monitor Driver  Battery Driver }
```

Current-dependent throttling

If the battery driver requires throttling based on both temperature and electrical current, the ACPI algorithm in thermal manager is no longer adequate because it cannot take current into account. To replace this algorithm, you must provide a policy driver that contains a custom algorithm, and load this driver on top of the driver stack for the thermal zone. This policy driver treats both the temperature sensor and current sensor as inputs, and arrives at a thermal policy based on the custom algorithm. Note that this thermal policy must operate within the capabilities of the thermal zone hardware. The policy is sent to the thermal manager, which updates logs and updates the thermal zone. The thermal zone then sends requests to the battery driver via the thermal cooling interface.

The following block diagram shows a policy driver that controls both the temperature and current of a battery device. The policy driver implements a custom algorithm in place of the thermal manager's algorithm. Unlike the thermal manager's algorithm, the custom algorithm takes both temperature and current into account.



Thermal management requirements

Hardware requirements

The following points are required for a good thermal hardware design:

- All systems meet applicable industry standards (for example, IEC 62368) for consumer electronics safety.
- Hardware must have fail-safe temperature trip point that shuts down the system or prevents boot.
- Sensor hardware must be accurate to +/- 2°C.
- Sensor hardware must not require software polling to determine that a threshold temperature has been exceeded.
- While operating, the system display brightness is never thermally limited to less than 100 nits.
- Battery charging is not throttled while either:
  - The system is idle and within the ambient temperature range below 35°C, or
  - The ambient temperature is below 25°C under any conditions.

HCK test requirements for modern standby PCs

All modern standby PCs must meet certain thermal requirements regardless of processor architecture and form factor. These requirements are tested for in the HCK:

- All modern standby PCs must have at least one thermal zone.
- Each thermal zone must report actual temperature on the sensor.
- At least one thermal zone must have critical shutdown temperature defined. An exception is made for Intel Dynamic Platform and Thermal Framework (DPTF).
- All modern standby PCs with fans must expose fan activity to the operating system.
- The fan needs to notify the operating system of its activity at all times, including during idle resiliency in modern standby. Currently, these notifications cause no action from the operating system. The main purposes for these notifications are diagnosability and telemetry. The fan notification can be integrated with existing tracing tools, including Windows Performance Analyzer. System designers can use these tools to tune the platform design.
- All modern standby PCs with fans must keep the fan off while in modern standby, the system "sleep" state.
- The HCK test here runs a realistic modern standby workload that should not cause the fan to turn on. During the transition to modern standby, the fan is allowed to stay on for up to 30 seconds from the time the display turns off.

For more information about the HCK tests, see Check Thermal Zones.

To run the HCK tests, do the following:

- First, enter this command to install the button driver:

  ```
  >>Button.exe -i
  ```

- To run all thermal tests for a PC with a fan, enter this command:

  ```
  >>RunCheckTz.cmd all
  ```

- To run all thermal tests for a PC without a fan, enter this command:

  ```
  >>RunCheckTz.cmd nofan all
  ```

Thermal management solutions

The Window thermal framework based on ACPI is the recommended thermal management solution for all systems. The primary benefits include the ability to easily diagnose thermal issues with inbox tools, and the ability to gather valuable telemetry in the field.

However, alternative solutions to the Windows thermal framework are acceptable if the above requirements are met. Core silicon and SoC vendors may have their own proprietary thermal solutions that are compatible with and supported by Windows—for example, implementations based on Intel Dynamic Platform and Thermal Framework (DPTF) for x86 processors and PEP implementations on Arm.

Diagnostics

To help system designers diagnose and evaluate system thermal behavior, Windows provides the following inbox and stand-alone tools.

Event logs

Windows records important thermal information in the event logs. This information can be used to quickly triage thermal conditions on any PC running Windows 8 or later without the need for additional tracing or tools. The following table contains the full list.

Channel	Source	ID	Event description
Windows Logs\System	Kernel power	125	**ACPI thermal zone being enumerated.** Windows logs this event during boot for every thermal zone.
Windows Logs\Systems	Kernel power	86	**The system was shut down due to a critical thermal event.** After a critical shutdown, Windows logs this event. This event can be used to diagnose whether a thermal critical shutdown has happened and to identify the thermal zone that caused the shutdown.

Channel	Source	ID	Event description
Applications and Services Logs\Microsoft\Windows\Kernel-Power\Thermal-Operational	Kernel power	114	**One thermal zone has engaged or disengaged passive cooling.** Windows logs this event when thermal throttling engages and disengages. This event can be used to confirm whether thermal throttling has happened and for which zones. This is helpful when triaging performance problems.

Critical event notification

In the event of a critical shutdown or hibernate caused by thermal conditions, the operating system must be notified of the event so that it can be recorded in the system event log. There are two ways to notify the operating system when this occurs:

- Use the ACPI thermal zone _CRT or _HOT method to automatically log the critical thermal event correctly. No additional work is needed other than to define a _CRT or _HOT value.
- For all other thermal solutions, the driver can use the following thermal event interface, which is defined in the Procpowr.h header file:

```
#define THERMAL_EVENT_VERSION 1 typedef struct _THERMAL_EVENT { ULONG
Version; ULONG Size; ULONG Type; ULONG Temperature; ULONG
TripPointTemperature; LPWSTR Initiator;  } THERMAL_EVENT,
*PTHERMAL_EVENT;  #if (NTDDI_VERSION >= NTDDI_WINBLUE) DWORD
PowerReportThermalEvent ( _In_ PTHERMAL_EVENT Event ); #endif
```

  The **PowerReportThermalEvent** routine notifies the operating system of a thermal event so that the event can be recorded in the system event log. Before calling **PowerReportThermalEvent**, the driver sets the members of the **THERMAL_EVENT** structure to the following values.

Version
THERMAL_EVENT_VERSION
Size
sizeof(THERMAL_EVENT)
Type
One of the THERMAL_EVENT_*XXX* values from Ntpoapi.h.
Temperature
The temperature, in tenths of a degree Kelvin, that the sensor was at after crossing the trip point (or zero if unknown).
TripPointTemperature
The temperature, in tenths of a degree Kelvin, of the trip point (or zero if unknown).
Initiator
A pointer to a NULL-terminated, wide-character string that identifies the sensor whose threshold was crossed.

```
The following thermal event types are defined in the Ntpoapi.h header file:

```
// // Thermal event types // #define THERMAL_EVENT_SHUTDOWN     0 #define
THERMAL_EVENT_HIBERNATE    1 #define THERMAL_EVENT_UNSPECIFIED  0xffffffff
```

Hardware platforms should use the thermal event interface only if thermal
solutions other than Windows thermal management framework are used. This
interface allows the operating system to gather information when a critical
shutdown occurs due to thermal reasons.
```

Performance counters

Performance counters offer a real-time information on the thermal behavior of the system. The following three pieces of data are polled for each thermal zone.

Thermal zone information

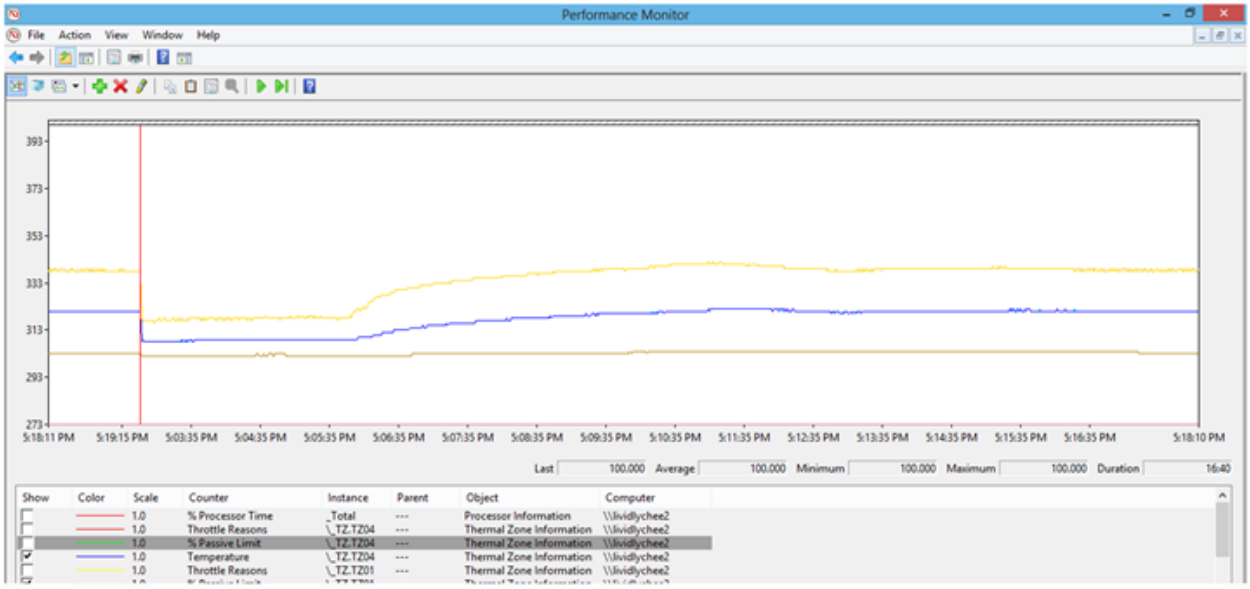
- **Percent passive limit**– The percentage throttled. 100 percent means the zone is not throttled.
- **Temperature**– The temperature of the thermal zone in degrees Kelvin.
- **Throttle reason**– The reason why a zone is being throttled:
  - 0x0 – The zone is not throttled
  - 0x1 – The zone is throttled for thermal reasons.
  - 0x2 – The zone is throttled to limit electrical current.

This information is polled only when requested — for example, by Windows Performance Monitor or by the typeperf command-line tool.

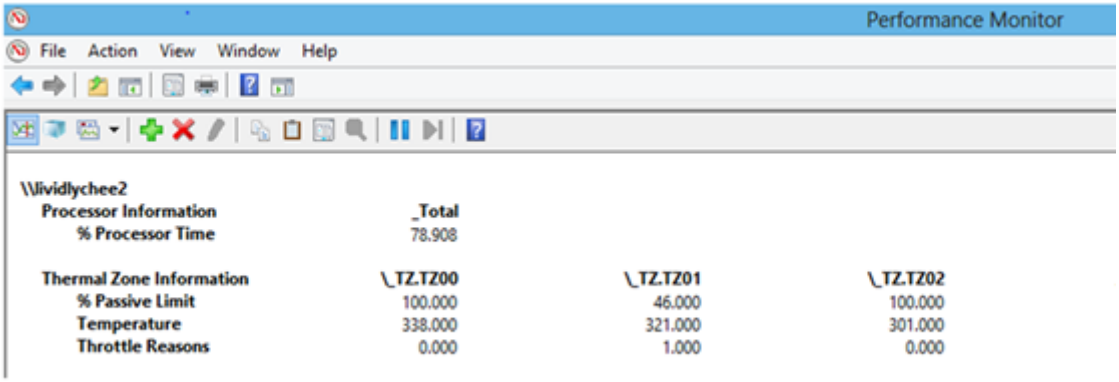
For more information about performance counters in general, see Performance Counters.

Performance monitor

Performance Monitor is a built-in application for polling and visualizing information. Performance Monitor can be a very powerful tool for comparing thermal conditions for system thermal designs. The following two sample screenshots show Performance Monitor in action when the fishbowl demo is run in Internet Explorer. In the first screenshot, Performance Monitor shows the increase in temperature of three thermal zones over time.



In the second screenshot, Performance Monitor reports the current throttle percentage, temperature, and reason for throttling.


Performance Monitor
File Action View Window Help
\\lividlychee2
Processor Information _Total
% Processor Time 78.908
Thermal Zone Information _TZ.TZ00 _TZ.TZ01 _TZ.TZ02
% Passive Limit 100.000 46.000 100.000
Temperature 338.000 321.000 301.000
Throttle Reasons 0.000 1.000 0.000

For more information, see Using Performance Monitor.

Windows Performance Analyzer (WPA)

As part of the ADK, Windows provides the Windows Performance Toolkit (WPT) for software tracing and analyzing. Inside WPT, system designers can use Windows

Performance Analyzer (WPA) to visualize the software traces and analyze the thermal behavior. For more information about how to install and use WPA, see Windows Performance Analyzer (WPA).

Providers

Include "Microsoft-Windows-Kernel-ACPI" to log events for temperature, thermal zone activity, and fan activity.

Include "Microsoft-Windows-Thermal-Polling" to enable polling of the temperature on each thermal zone. If this is not included, temperatures will only be reported when they rise above the passive and/or active trip points. The period of polling can be controlled by specifying a flag to the provider.

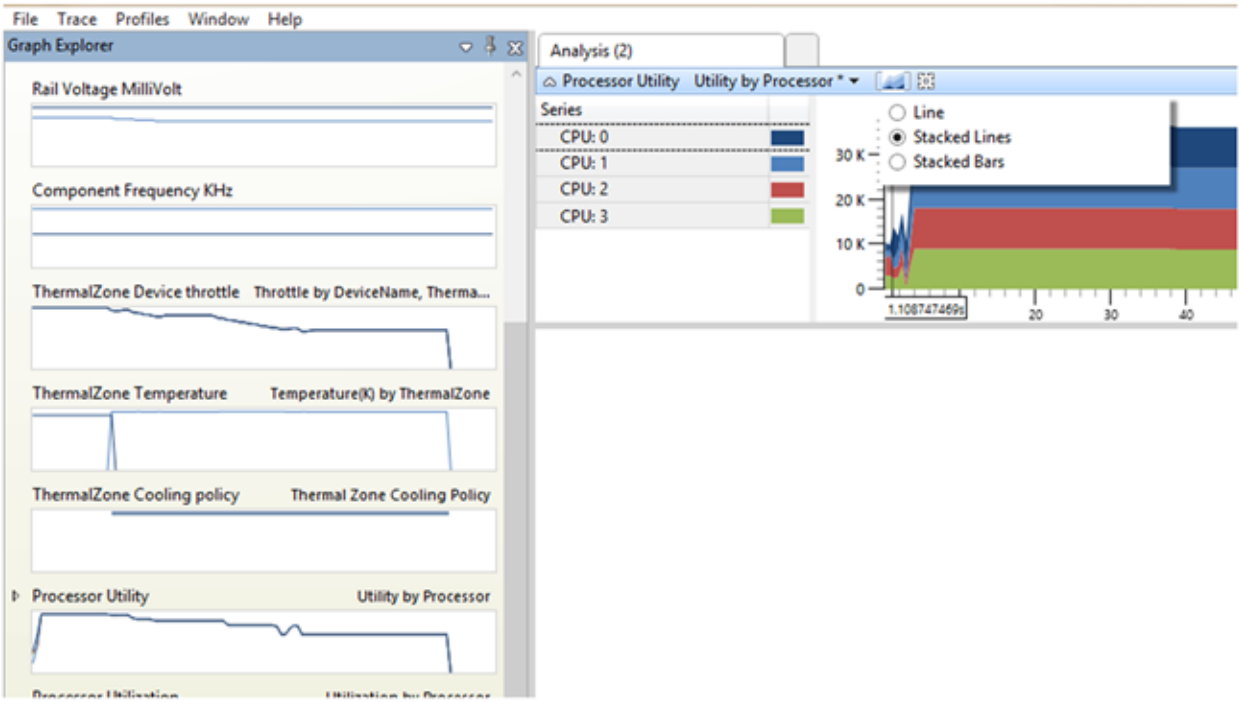
Flag	Period of polling
None	1 second
0x1	1 second
0x2	5 seconds
0x4	30 seconds
0x8	5 minutes
0x10	30 minutes

Processor utility

Before digging into thermal throttling data, it's a good idea to double-check the processor utility information to make sure that the processor utility pattern is consistent with what the workload should be. To confirm that the workload is setup correctly, follow these steps:

1. Open the ETL file with the WPA tool.
2. In the **Graph Explorer**, select **Power**, and then **Processor Utility**.
3. Change the **Graph Type** to **Stacked Lines**.

The following screenshot shows the **Processor Utility** graph.

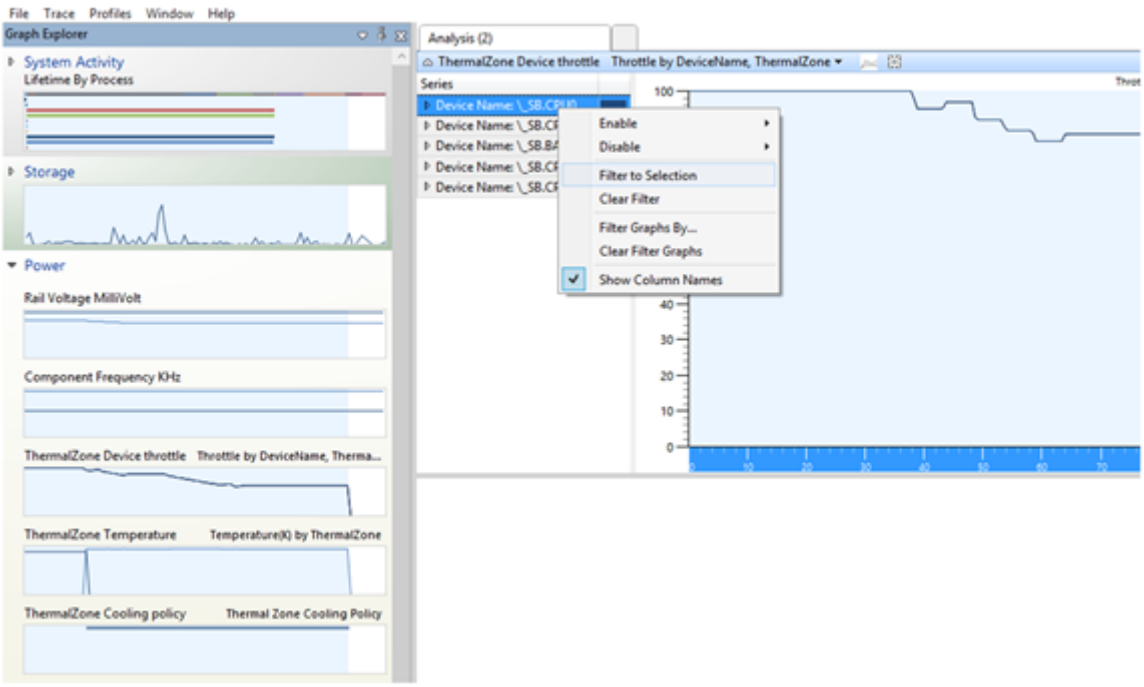


Thermal zone throttling percentage

When a thermal zone is throttling, the software tracing file logged all the thermal throttling percentage changes, temperature changes, and cooling policy changes. To view the information in the tracing file, follow these steps:

1. Open the ETL file using the WPA tool.
2. In the **Graph Explorer**, select **Power**, and then **ThermalZone Device Throttle**.
3. You can select interesting devices through **Applying Filtering**.

The following screenshot shows the **ThermalZone Device Throttle** graph and filtering options.

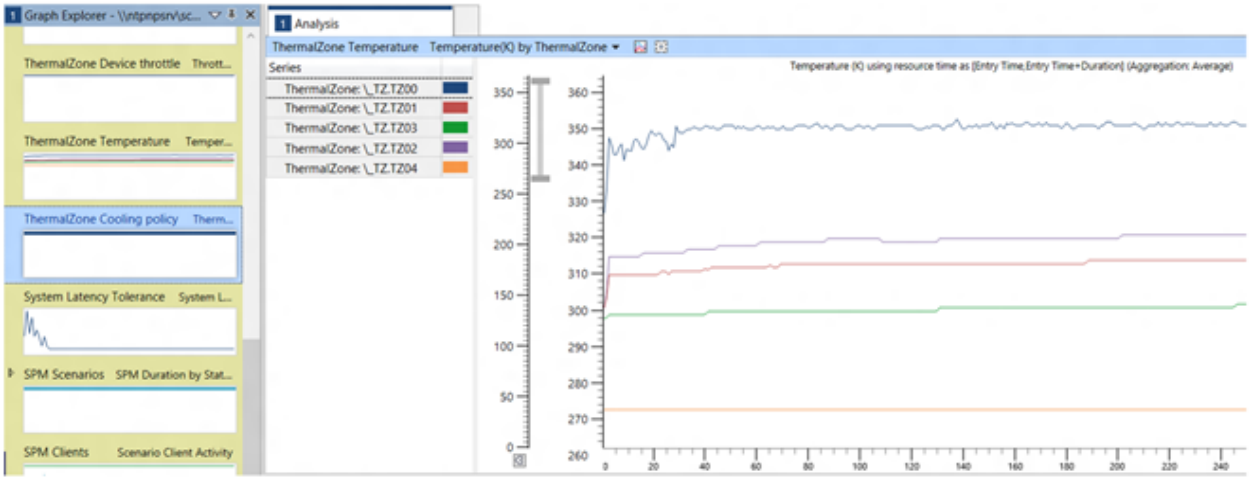


Thermal zone temperature

Using performance counter information, the temperature of the system can also be monitored while no throttling is engaged. Follow these steps:

1. Enable the desired providers while taking a trace.
2. Make sure performance counters are still being polled (Performance Monitor is still running). For more information, see Performance counters.
3. Open the ETL file using the WPA tool.
4. In the **Graph Explorer**, select **Power**, and then **Temperature (K) by ThermalZone**.
5. You should see temperature over time for each thermal zone.

The following screenshot shows a graph of temperature over time for five thermal zones.



Thermal Management Datatypes

Article • 01/18/2019

This topic discusses the datatypes that are used for Thermal Management in Windows.

In this section

Topic	Description
THERMAL_READ_POLICY	The THERMAL_POLICY struct is used with the IOCTL_THERMAL_READ_POLICYinput/output (I/O) control code.
THERMAL_WAIT_READ	The THERMAL_WAIT_READ struct is used as the input buffer by the IOCTL_THERMAL_READ_TEMPERATURE IOCTL control code.

Related topics

Thermal Management IOCTLs

THERMAL_POLICY

Article • 01/18/2019

The THERMAL_POLICY struct is used with the IOCTL_THERMAL_READ_POLICY IO control code. Policy drivers use IOCTL_THERMAL_READ_POLICY to set the thermal policy for their associated thermal zone. If no thermal policy is specified, then the kernel uses the policies specified by the ACPI objects and methods declared under the thermal zone, as described in the ACPI specification.

Syntax

C++

```
typedef struct _THERMAL_POLICY {

ULONG Version; BOOLEAN WaitForUpdate;

BOOLEAN Hibernate;

BOOLEAN Critical;

BOOLEAN ThermalStandby;

ULONG ActivationReasons;

ULONG PassiveLimit;

ULONG ActiveLevel;

} THERMAL_POLICY, *PTHERMAL_POLICY;
```

Members

Version

Specifies the version of the policy structure. The current version is THERMAL_POLICY_VERSION_1.

WaitForUpdate

Specifies if the policy driver should complete the IOCTL_THERMAL_READ_POLICY IOCTL immediately. If the state is up to date and WaitForUpdate is TRUE, the policy driver should pend the IOCTL in a cancellable state until one of its policy preferences has changed, at which point it should complete the IOCTL with the new preferences. If the

state is not up to date or WaitForUpdate is FALSE, the policy driver should immediately complete the IOCTL with the new preferences.

Hibernate

The policy driver specifies TRUE to indicate that the system should hibernate to address a thermal condition. Once the thermal condition has been resolved, the policy driver should set this member to FALSE. If the system does not support hibernate or if hibernate is disabled, the system will shut down instead.

Critical

The policy driver specifies TRUE to indicate that the system should shut down to address a thermal condition.

ThermalStandby

The policy driver specifies TRUE to indicate that the system should enter standby to address a thermal condition. Once the thermal condition has been resolved, the policy driver should set this member to FALSE.

ActivationReasons

Specifies why the thermal zone is throttled.

- 0x00000000

  Indicates that the zone isn't throttled.

- #define TZ_ACTIVATION_REASON_THERMAL 0x00000001

  Indicates that the zone is throttled for thermal reasons.

- #define TZ_ACTIVATION_REASON_CURRENT 0x00000002

  Indicates that the zone is throttled because the system cannot deliver sufficient current.

PassiveLimit

Specifies the amount by which the thermal zone should be throttled. 100 represents unthrottled while 0 represents fully throttled.

ActiveLevel

Specifies the index (from 0 through 9) of the current set of active cooling devices that should be engaged. For example, setting an ActiveLevel of 4 would engage all devices in

the thermal zone's _AL4 device list, as well as any higher-indexed device lists (_AL5, etc.). Specifying 10 will disable all active cooling devices.

Related topics

THERMAL_WAIT_READ

Thermal management in Windows

THERMAL_WAIT_READ structure

Article • 01/18/2019

The **THERMAL_WAIT_READ** struct is used as the input buffer by the IOCTL_THERMAL_READ_TEMPERATURE IOCTL control code.

Syntax

C++

```
typedef struct _THERMAL_WAIT_READ {

ULONG Timeout;

ULONG LowTemperature;

ULONG HighTemperature;

} THERMAL_WAIT_READ, *PTHERMAL_WAIT_READ;
```

Members

Timeout

The time to wait before returning temperature data, in milliseconds. 0 indicates the temperature should be read immediately, with no wait. -1 indicates the read should not time out.

LowTemperature

The lower threshold for returning the new temperature given in tenths degrees Kelvin. As soon as the temperature drops below the low temperature threshold, the driver should complete the IOCTL. If the temperature is already below the low temperature, the IOCTL should be completed immediately.

HighTemperature

The upper threshold for returning the new temperature given in tenths degrees Kelvin. As soon as the temperature rises above the high temperature threshold, the driver should complete the IOCTL. If the temperature is already above the high temperature, the IOCTL should be completed immediately.

Related topics

Thermal Management in Windows

Thermal Management Datatypes

Thermal Management IOCTLs

Thermal Management IOCTLs

Article • 01/18/2019

This topic discusses the input/output controls (IOCTLs) that are used for Thermal Management in Windows.

Driver implementers need to implement responses to the following input/output (I/O) control codes.

In this section

Topic	Description
IOCTL_THERMAL_READ_POLICY	The IOCTL_THERMAL_READ_POLICY I/O control request is sent by the kernel to a thermal zone's policy driver to read the drivers' preferred policy.
IOCTL_THERMAL_READ_TEMPERATURE	The IOCTL_THERMAL_READ_TEMPERATURE I/O control request is sent by the ACPI driver to a temperature sensor driver. Upon receiving this request, the driver might be required to provide the temperature of the thermal zone.

IOCTL_THERMAL_READ_POLICY

Article • 11/05/2020

The IOCTL_THERMAL_READ_POLICY input/output (I/O) control request is sent by the kernel to a thermal zone's policy driver to read the drivers' preferred policy.

Input Parameters

To retrieve the input buffer, call **WdfRequestRetrieveInputBuffer**. The input buffer contains a THERMAL_POLICY struct that specifies the current policy that the kernel has for the thermal zone.

Output Parameters

To retrieve the output buffer, call **WdfRequestRetrieveOutputBuffer**. The *Buffer* parameter points to a THERMAL_POLICY structure. Set the THERMAL_POLICY's members to your policy driver's preferred thermal policy.

I/O Status

Complete the request by calling **WdfRequestCompleteWithInformation** and set Status to STATUS_SUCCESS if the request is successful. Otherwise, an appropriate NTSTATUS error condition. The driver can pass the bytes returned to `sizeof(ULONG)` in the **Information** parameter.

Related topics

IOCTL_THERMAL_READ_TEMPERATURE

Thermal management in Windows

IOCTL_THERMAL_READ_TEMPERATURE

Article • 11/05/2020

The IOCTL_THERMAL_READ_TEMPERATURE input/output (I/O) control request is sent by the ACPI driver to a temperature sensor driver. Upon receiving this request, the driver might be required to provide the temperature of the thermal zone.

Input Parameters

To retrieve the input buffer, call **WdfRequestRetrieveInputBuffer**. The input buffer contains a THERMAL_WAIT_READ structure. The Timeout member indicates when to return to the thermal zone's temperature after which the request will expire. If the received **Timeout** value is -1, the request never expires.

Output Parameters

To retrieve the output buffer, call **WdfRequestRetrieveOutputBuffer**. The Buffer parameter points to a **ULONG**. Set the value to the thermal zone temperature, in tenths degree Kelvin.

I/O Status

Complete the request by calling **WdfRequestCompleteWithInformation** and set Status to STATUS_SUCCESS if the request is successful. Otherwise, an appropriate NTSTATUS error condition. The driver can pass the bytes returned to sizeof(ULONG) in the Information parameter. Related topics Thermal management in Windows

Related topics

IOCTL_THERMAL_READ_POLICY

Thermal management in Windows

Thermal Management Functions

Article • 12/12/2024

This topic discusses the API functions are used for Thermal Management in Windows.

In this section

Expand table

Topic	Description
PoCreateThermalRequest	The PoCreateThermalRequestCreates a new thermal request object.
PoDeleteThermalRequest	The PoDeleteThermalRequestDeletes a power request object.
PoGetThermalRequestSupport	The PoGetThermalRequestSupportQueries whether the target object supports passive and active cooling.
PoSetThermalActiveCooling	The PoSetThermalActiveCoolingSets the active cooling of target object.
PoSetThermalPassiveCooling	The PoSetThermalPassiveCoolingSets the passive cooling of target object.

PoCreateThermalRequest function

Article • 12/10/2024

The **PoCreateThermalRequest** creates a new thermal request object.

Syntax

```
C++

NTKERNELAPI
NTSTATUS
PoCreateThermalRequest (
    PVOID *ThermalRequest,
    PDEVICE_OBJECT TargetDeviceObject,
    PDEVICE_OBJECT PolicyDeviceObject,
    PCOUNTED_REASON_CONTEXT Context,
    ULONG Flags
    )
```

Parameters

`ThermalRequest`

Supplies a pointer to receive the thermal request object.

`TargetDeviceObject`

Supplies the device object of the device being cooled.

`PolicyDeviceObject`

Supplies the thermal device object of the device making the cooling request.

`Context`

A pointer to a COUNTED_REASON_CONTEXT structure that describes why the caller is creating the thermal request object.

`Flags`

Supplies flags that modify the behavior of the thermal request.

Return Value

PoCreateThermalRequest returns STATUS_SUCCESS if the call is successful. If the call fails, possible error return codes include the following:

Expand table

Return code	Description
STATUS_INVALID_PARAMETER	If any of the input parameters are **NULL**
STATUS_INSUFFICIENT_RESOURCES	There is not enough memory available to create a power request object

Remarks

This routine creates a Thermal request object. One thermal request needs to be created per device that needs to be cooled. After creation the caller can use **PoGetThermalRequestSupport** to query cooling (active or passive) cooling capabilities of the device, and use **PoSetThermalPassiveCooling** and **PoSetThermalActiveCooling** routines to make cooling requests on the device.

When the thermal request object is no longer needed, the caller must delete the object by calling the PoDeleteThermalRequest routine. The driver must delete the thermal request object before it deletes the device object that was used to create the thermal request object.

Requirements

Expand table

Minimum supported client	Available starting with Windows 10 1703 release.
Target Platform	Universal
Header	wdm.h (include Wdm.h, Ntddk.h, Ntifs.h)
Library	NtosKrnl.lib
DLL	NtosKrnl.exe
IRQL	<= APC_LEVEL

PoDeleteThermalRequest function

Article • 12/10/2024

This **PoDeleteThermalRequest** cleans up an existing thermal request object.

Syntax

```
C++

NTKERNELAPI
VOID
PoDeleteThermalRequest (
    PVOID ThermalRequest
    )
```

Parameters

`ThermalRequest`

A pointer to a thermal request object that was created by the PoCreateThermalRequest routine.

Return Value

None

Remarks

The driver must delete the thermal request object before it deletes the device object that was used to create the thermal request object.

Requirements

Expand table

Minimum supported client	Available starting with Windows 10 1507 release.

Target Platform	Universal
Header	ntifs.h (include Wdm.h, Ntddk.h, Ntifs.h)
Library	NtosKrnl.lib
DLL	NtosKrnl.exe
IRQL	<=APC_LEVEL

PoGetThermalRequestSupport function

Article • 12/10/2024

The **PoGetThermalRequestSupport** returns whether or not the thermal request supports the specified cooling mechanism.

Syntax

```
C++

NTKERNELAPI
BOOLEAN
PoGetThermalRequestSupport (
    PVOID ThermalRequest,
    PO_THERMAL_REQUEST_TYPE Type
    )
```

Parameters

`ThermalRequest`

Supplies a pointer to receive the thermal request object.

`Type`

Supplies the cooling mechanism to check.

Return Value

TRUE if the specified cooling mechanism is supported, FALSE otherwise.

Remarks

This routine can used to check if a particular device provides support for active or passive cooling.

Requirements

Expand table

Minimum supported client	Available starting with Windows 10 1703 release.
Target Platform	Universal
Header	wdm.h (include Wdm.h, Ntddk.h, Ntifs.h)
Library	NtosKrnl.lib
DLL	NtosKrnl.exe
IRQL	<= APC_LEVEL

PoSetThermalActiveCooling function

Article • 12/10/2024

The **PoSetThermalActiveCooling** updates the active cooling constraint associated with the thermal request, propagating the update to the device's cooling interface if needed.

Syntax

```
C++

NTKERNELAPI
NTSTATUS
PoSetThermalActiveCooling (
    PVOID ThermalRequest,
    BOOLEAN Engaged
    )
```

Parameters

`ThermalRequest`

Supplies a pointer to receive the thermal request object.

`Engaged`

Supplies TRUE if active cooling should be engaged, FALSE otherwise.

Return Value

NTSTATUS

Requirements

Expand table

Minimum supported client	Available starting with Windows 10 1703 release.
Target Platform	Universal

Header	wdm.h (include Wdm.h, Ntddk.h, Ntifs.h)
Library	NtosKrnl.lib
DLL	NtosKrnl.exe
IRQL	<= APC_LEVEL

PoSetThermalPassiveCooling function

Article • 12/10/2024

The **PoSetThermalPassiveCooling** updates the passive cooling constraint associated with the thermal request, propagating the update to the device's cooling interface if needed.

Syntax

```
C++

NTKERNELAPI
NTSTATUS
PoSetThermalPassiveCooling (
    PVOID ThermalRequest,
    UCHAR Throttle
    )
```

Parameters

`ThermalRequest`

Supplies a pointer to receive the thermal request object.

`Throttle`

Supplies the new passive cooling throttle level.

Return Value

NTSTATUS

Requirements

Expand table

Minimum supported client	Available starting with Windows 10 1703 release.
Target Platform	Universal

Header	wdm.h (include Wdm.h, Ntddk.h, Ntifs.h)
Library	NtosKrnl.lib
DLL	NtosKrnl.exe
IRQL	<= APC_LEVEL

Versatile input platform

Article • 01/18/2019

Precision touchpad

While touch provides the best experience for directly manipulating user interfaces and lowers the barrier for content consumption, it is not optimized for all productivity needs. A goal of Windows is to optimize each input mechanism for its strengths, taking advantage of the touchpad's natural precision and other attributes. Precision Touchpads are designed to be used alone or paired with touch, and they are designed to meet users' growing expectations of the experience on their PCs.

Consistent and reliable

When users sit down at a new PC, the touchpad should work in a familiar and expected fashion. The goal of PTP is to bring that consistency of experience to Windows devices, yet at the same time allow for the exciting breadth of form factors that exist in our ecosystem. Touchpad settings have been added to the modern control panel, providing centralized control in an easy to locate place. Moreover, devices should just work. Users and enterprises do not want to have to find the right driver to get basic functionality. Reliability should be architected for from the ground up. Precision touchpads require no drivers to update, and the experience works out of the box every time.

Familiar, fast, and responsive

Windows 10 PTP leveraged the Windows 8 touch language where it made sense, providing a familiar set of gestures to users. One of the large investments in Windows 8 touch was in DirectManipulation, the component that powers the buttery smooth pan and zoom experience for touch. That same component is utilized with PTP to provide an amazing pan and zoom experience for touchpads that apps get for free and that competitors cannot match. The strengths of touchpads have been optimized – for example, where touch is required to be "stick to your finger", because touchpads are indirect input devices, there is more flexibility and it can be equally easy to pan very precisely or very quickly to get to the end of a long document. Additionally, the are ecosystem investments in hardware, raising the bar for touchpads. Precision touchpads are more accurate, higher resolution, and lower latency than other devices, and come paired with excellent industrial design.

An experience you don’t want to turn off

One of the top complaints from users is accidental activation. While typing, suddenly a brush of the touchpad will change focus to another app, or the cursor moves and or the text gets chopped up, or worse. Windows 10 PTP invests heavily in addressing this problem and ensures the touchpad never gets in the way of what is trying to be accomplished. Addressing this problem permeates all PTP work and includes efforts in avoiding accidental taps, edge gestures, palms and thumbs resting on the touchpad, and more.

A strong guarantee through hardware compatibility

Precision Touchpads are backed with a strong hardware compatibility program that guarantees a high quality experience matched with great hardware. The device requirements ensure hardware features superior accuracy, resolution, latency, and report rate characteristics. These attributes make the user experience shine. Additionally, Windows collaborated closely with hardware partners around the design and materials used for the experience. The result is an unparalleled integration between materials, hardware engineering, and software platform.

Building best-in-class Wireless Projection solutions with Windows 10

Article • 10/16/2019

Overview

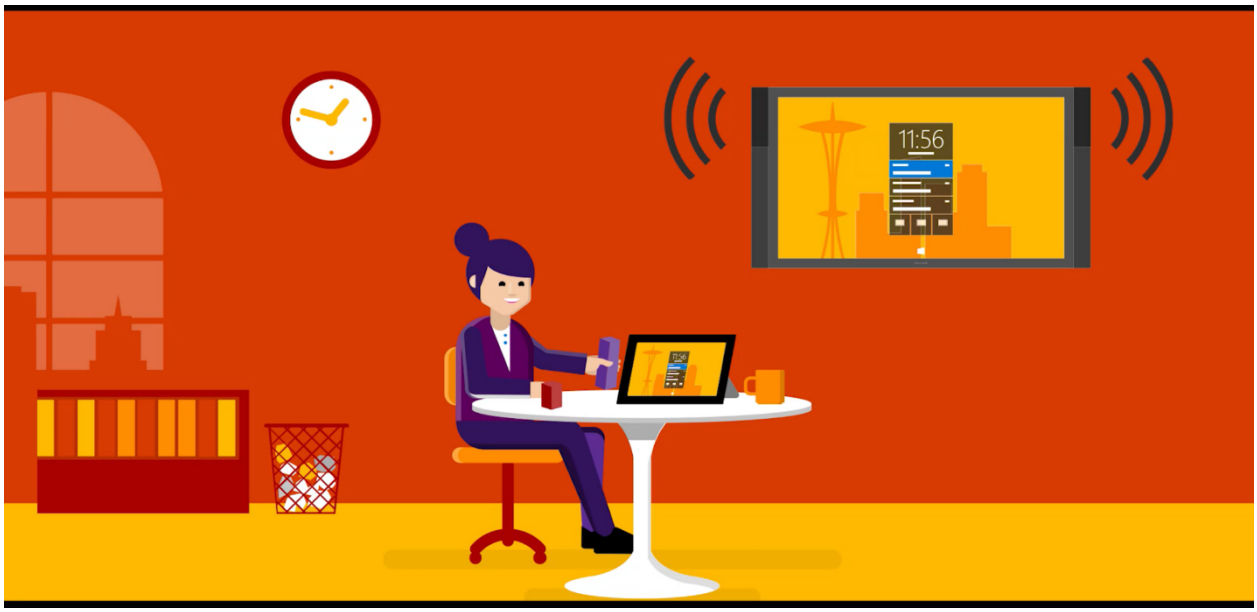
Wireless Projection enables a user to mirror their screen content to a wireless receiver such as a TV or media device. This creates the ability to project content from a PC to a larger screen for the purposes of sharing and collaboration.

In order to provide Windows users with seamless wireless projection experiences, Windows 10 relies on hardware, software, and firmware from its hardware partners. Quality Wi-Fi, Graphics, Composition, Media encoders/decoders and receiver components are fundamental to this experience.

This guide was written to help Wi-Fi IHVs, Graphics IHVs, PC OEMs and Miracast receiver manufacturers:

- Understand what wireless projection components are provided as part of Windows 10
- Understand what wireless projection capabilities Windows 10 supports
- Understand what features and functionality need to be implemented
- Select the best hardware for your product
- Implement the various functionality listed herein
- And lastly, provide methods to validate your implementation

All of this is provided to help create a best-in-class wireless projection experience with Windows 10 and your products.


11:56

Understanding Windows 10 Wireless Projection

Article • 03/18/2021

Windows 10 provides a seamless wireless projection experience. As you are building your part of the wireless projection solution it is important to understand what that functionality includes.

The Windows 10 Wireless Projection User Interface

First and foremost, Windows provides a native connection experience that allows a user to connect to a wireless receiver. A connection to a wireless receiver can be made in several ways:

- **Through the Action Center.** In the Action Center (Figure 1), click the **Connect Quick Action**.
- **Using a hotkey.** Select the **Windows logo key+ K** (Figure 2).
- **With the Device Picker UI**. Windows Apps that support casting contain a device picker UI, such as the **Cast to device** feature in the Windows Movie & TV App (Figure 3) or the Edge browser.

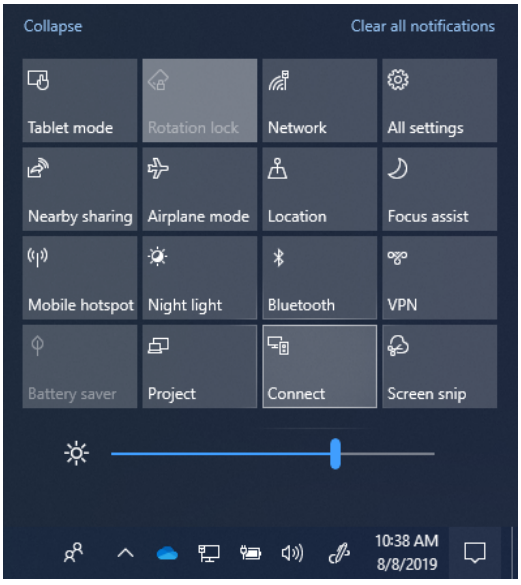


Figure 1: Connect Quick Action

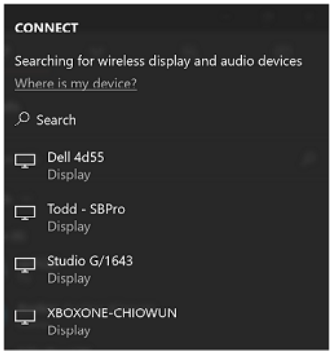


Figure 2: Connect UI

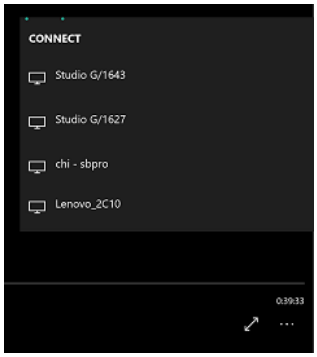


Figure 3: Device Picker UI

Windows 10 supports and manages two different methods for creating a wireless projection stream. Both methods are handled behind the scenes by Windows and leverage the exact same UI shown above.

Wireless Projection using Miracast

Support for Miracast has been present since the first Windows 10 version, and since that time Windows investments in Miracast as a projection experience have only increased.

Wireless Projection over Miracast offers several benefits:

- A simple connection experience that allows a user to find and connect to Miracast receivers.
- Implementation of the Miracast standard to ensure interoperability with hundreds of millions of Miracast devices.
- A native RTSP stack fine-tuned to work for Miracast, requiring no additional software is required outside the Windows 10 OS.
- Support for UIBC (User Input Back Channel), which allows inputs from the Miracast receiver (touch, stylus, mouse, keyboard and gamepad) to control the Miracast sender, if---and only if---the user explicitly allows this.
- High quality interoperability with industry leading Microsoft Miracast receivers (Microsoft Wireless Display Adapter) as well as leading 3rd party Miracast receivers.
- Support for the projection of Protected Content (if HDCP keys are present).
- Support for the use of a PIN based pairing when connecting to a Miracast receiver.
- Persistent profiles, remembering if you have connected to a specific Miracast receiver in the past. The ability to remember a profile for reconnecting to a Miracast receiver reduces the time it takes to connect on subsequent connections.
- Support for Miracast extensions that enable additional capabilities, dramatically improving the Miracast experience.

Wireless Projection over an existing Wi-Fi network

It has been observed that in 90% of cases, when a User starts a Wireless Projection stream, the device they are using is already connected to an existing Wi-Fi network,

either in a home or a business. In response to this, Microsoft has extended the ability to send a Miracast stream over a local network rather than over a direct wireless link in the Windows 10 Creators Update.

Wireless Projection over an existing Wi-Fi network offers several benefits:

- This solution leverages an existing connection which can significantly reduce the time it takes to project content.
- The PC does not need to manage two simultaneous connections (a Wi-Fi connection and a Wi-Fi Direct connection to the receiver), each of which would only get half the maximum bandwidth.
- Using an existing connection simplifies the work that the wireless device needs to perform, which both increases reliability and provides a very stable stream.
- The user does not have to change how they connect to a receiver because they use the same UX (as is shown in Figures 2-4).
- Windows will only choose to project over an existing connection if the connection is trusted, either over an Ethernet or a secure Wi-Fi network.
- There are no required changes to the wireless drivers or hardware on a PC.
- This also works well with older wireless hardware that is not optimized for Miracast over Wi-Fi Direct.
- Windows automatically detects when a receiver supports this functionality and will send the video stream over the existing network path when applicable.

Considerations and Requirements for PC manufacturers

Article • 11/05/2020

Wireless projection generates a stream of data that needs to be quickly created, wirelessly delivered and rendered in real-time. PC Manufacturers must thoughtfully select a number of Wi-Fi, Graphics, and Audio component characteristics that directly affect the quality of the wireless projection experience. Consider the following when approving components:

Wi-Fi device recommendations

Wi-Fi device band selection: Windows 10 can perform Miracast over both the 2.4GHz and 5GHz bands and works well with an 802.11n or 802.11ac capable Wi-Fi adapter. The 5GHz band, however, offers higher bandwidth and more channels with less interference, ideal conditions for a higher quality stream. Therefore, solutions supporting multi-band are recommended.

Wi-Fi Antennae configuration: Typically, an 802.11ac radio using the 5GHz band and a 2x2 antennae configuration produces a higher quality Miracast experience than any 802.11n or 1x1 combination. Solutions supporting 2x2 antenna configurations are recommended.

Wi-Fi Driver Type: Ensure that Wi-Fi adapter manufacturers provide drivers specifically designed for the Windows 10 Wireless Driver Interface (WDI). Wireless drivers developed for the WDI framework are precisely designed for Miracast usage. These drivers successfully demonstrate higher connection rates and yield high-quality Miracast sessions.

Wi-Fi Driver verification: Lastly, verify that Wi-Fi device manufacturers have successfully passed all the tests within the following Windows Hardware Lab Kit (HLK) test suites:

- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicDiscoveryTests.*
- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicPairingTests.*
- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicReconnectTests.*

Graphics device recommendations

Graphics Driver type: The Microsoft Miracast stack functions upon WDDM 2.0 (Window's Display Driver Model) or greater. WDDM 2.0 was introduced with Windows 10. A WDDM 2.0 or higher driver is required.

Hardware Accelerated Encode: A Miracast sender's graphics device/driver must support a hardware accelerated AVC (H264) Encoder. Note: HDCP 2.0 or greater is required to project protected content from most major video streaming content providers. Hardware accelerated AVC (H264) Encode is required.

Content Protection: A Miracast sender's graphics device/driver should support an HDCP 2.x transmit key. Note: HDCP 2.0 or greater is required to project protected content from most major video streaming content providers. Support for HDCP 2.x transmit key is recommended.

Hardware Accelerated High Resolution Encoding: Windows 10 supports both hardware accelerated AVC and HEVC. A Miracast sender's graphics device/driver must support hardware accelerated AVC encode and decode. Support for hardware accelerated HEVC (H265) encoding/decoding is recommended as it enables higher resolutions, for instance 4K, at higher bitrates while using less bandwidth than AVC.

Audio device recommendations

Audio Codec: An audio device/driver must support LPCM (Linear pulse code modulation) to maintain lower latency, Windows 10 only supports LPCM.

PC Manufacturer Checklist

Device Type	Component	Requirement	Checklist
Wi-Fi Device	Wi-Fi device band selection	802.11n 2.4 GHz band	Required (minimum)
		802.11n 5 GHz band	Recommended
		802.11ac 5 GHz band	Recommended
	Wi-Fi Antennae configuration	2x2 antennae	Recommended
	Wi-Fi Driver Type	WDI Driver model	Required
	Wi-Fi Driver verification	Microsoft Hardware Lab Kit (HLK)	Required
Graphics Device	Graphics Driver Type	WDDM 2.0 or higher driver	Required

Device Type	Component	Requirement	Checklist
	Hardware Accelerated Encode	Hardware accelerated AVC (H264) Encode	Required
	Hardware Accelerated Encode	Hardware accelerated HEVC (H265) Encode	Recommended
	Content Protection	Support for HDCP 2.x transmit key	Recommended
Audio device	Audio Codec	An audio device/driver must support LPCM	Required

Considerations and requirements for Wi-Fi device manufacturers (IHVs)

Article • 10/16/2019

Designing a best-in-class Microsoft Windows 10 Miracast experience requires Device Manufacturers (IHVs) to adhere to high quality design and driver development standards.

Wi-Fi Driver Type: Ensure that the Wi-Fi device driver adheres to the Wireless Driver Interface (WDI) model. Wireless drivers developed to the WDI framework are precisely designed for Miracast usage. These drivers successfully demonstrate higher connection rates and yield high-quality Miracast sessions.

Wi-Fi Driver Capabilities: Ensure that the Wi-Fi driver declares capability support for each of these functions through WDI:

- Wi-Fi Direct Device
- Wi-Fi Direct GO
- Wi-Fi Direct Client
- P2P Device Discovery
- Extended Channel Switch Announcement
- Miracast Sink
- P2P GO ports count
- P2P Clients Port Count

Wi-Fi Driver verification: The Wi-Fi driver must pass all the tests in the following Windows Hardware Lab Kit (HLK) test suites:

- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicDiscoveryTests.*
- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicPairingTests.*
- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicReconnectTests.*

Extended channel switch announcement (eCSA): In almost all cases where a Windows 10 device connects to a Miracast receiver it is also simultaneously connected to a Wi-Fi access point for Internet connectivity. In many cases like this the receiver and the Wi-Fi access point will be on different wireless channels. This type of multi-channel mode typically causes video quality to degrade, because the wireless device must continuously switch between channels to maintain simultaneous connectivity to both endpoints. In several cases like this, a Windows client can determine a multi-channel condition and can use eCSA to transform from Multi-Channel to Single Channel conditions.

Wi-Fi Device Manufacturer Checklist

Component	Requirement	Checklist
Wi-Fi device band selection	802.11n 2.4 GHz band	Required (minimum)
	802.11n 5 GHz band	Recommended
	802.11ac 5 GHz band	Recommended
Wi-Fi Antennae configuration	2x2 antennae	Recommended
Wi-Fi Driver Type	WDI Driver model	Required
Wi-Fi Driver Capabilities	Wi-Fi Direct Device, Wi-Fi Direct GO, Wi-Fi Direct Client, P2P Device Discovery, Extended Channel Switch Announcement, Miracast Sink, P2P GO ports count, and P2P Clients Port Count	Required
Wi-Fi Driver channel management	Extended channel switch announcement (eCSA)	Recommended

Validation	Validation Method	Checklist
Wi-Fi Driver verification	Microsoft Hardware Lab Kit (HLK)	Required
Wi-Fi Interoperability Testing	Wi-Fi Alliance Miracast Certification testing	Strongly Recommended
Wireless Projection Stress Testing	Microsoft Wireless Projection Stress Lab and Testing	Strongly Recommended

For more information about wireless projection verification methods, see Validating your implementation.

Considerations and Requirements for Receiver Manufacturers

Article • 10/16/2019

Miracast-specific features to support

The Windows client provides a standards conformant Miracast source implementation. In addition, we've added support for several extensions that improve the overall experience. Manufacturers building Miracast receivers should support the following features, extensions and techniques to deliver a high-quality Miracast experience.

Wi-Fi features

Implementation of WPS information element (IE) attributes for telemetry: Windows captures information about the success or failure of Miracast connections among the hundreds of Miracast receiver models in the market today. By setting a few simple attributes, Windows can identify the type of device Windows clients are connecting to. This allows us to reach out to you as a hardware partner if we notice that there is an incompatibility or sudden drop in quality.

We request that you fill in the following WPS Information Element (IE) attributes within WPS. The specific mappings that we use for Windows are:

- **WPS:Manufacturer**: Please put in the name of the Manufacturer, likely OEM or OEM + Chipset IHV, ex: Microsoft
- **WPS:Model**: This is the Unique Model for the device, ex: MWDA
- **WPS:Model-Number**: This is used to store the Firmware Version, ex: 1.3.8350

IP address allocation in EAPOL-key frames: Implementing IP addressing allocation within EAPOL-key frames reduces the amount of time taken to connect by including the IP address in the Wi-Fi Direct exchange itself. The PC connects more quickly by skipping the need to perform DHCP address assignment after the connection is established.

Extended channel switch announcement (eCSA): In almost all cases where a Windows 10 device connects to a Miracast receiver it is also simultaneously connected to a Wi-Fi access point for Internet connectivity. In many cases like this the receiver and the Wi-Fi access point are on different wireless channels. This type of multi-channel mode typically causes video quality to degrade; the wireless device must continuously switch

between channels to maintain simultaneous connectivity on both endpoints. The Windows client can sometimes determine a multi-channel condition and use eCSA to transform from Multi-Channel to Single Channel conditions.

Support for persistent profiles: Windows supports persistent P2P groups. When persistent P2P groups are supported on a receiver, Windows will perform a reconnection rather than a pairing and therefore connect more quickly to the receiver. If persistent profiles are not supported on the receiver, Windows 10 will perform a new pairing on each connection attempt, resulting in longer delay before streaming can begin.

TCP Retry tuning during connection: During the establishment of the RTSP session, keep TCP retry intervals very low. Performing exponential back-off early in the cycle results in long connect times if a single message must be retransmitted.

GC vs. GO determination: The Windows client will automatically adjust to its role during connection negotiation. Because Miracast uses a Wi-Fi Direct connection, Windows clients can assume one of several Wi-Fi Direct roles dynamically. It is important to note that while the Windows client will prefer to act as the Wi-Fi Direct GO (Intent 14) it is fully capable of acting as a GC or Auto-GO as well.

Direct Association: If the Miracast receiver is acting as a GO, a peer may attempt to reconnect to you. In this case you will receive an association request directly from that peer. If you no longer have a profile for that peer, fail the association with an association response frame which includes a P2P IE with status (8 - Fail; Unknown P2P Group); Windows 10 will retry the connection with a new pairing attempt.

UIBC tuning: If you plan to implement UIBC on a Miracast receiver, disable the Nagle algorithm since it queues frames before sending. Queueing cursor and keyboard frames results in latency, which is perceived by the user as sluggishness or delay. These types of packets should be sent as quickly as possible without queueing.

Projection over an existing Wi-Fi network: When a receiver supports connectivity over an infrastructure network the Windows 10 client will prefer that method of connection and establish its stream over the infrastructure. When projection over an existing W-Fi network is not supported or the connection attempt fails the Windows client will fall back to using traditional Miracast.

Graphics features

Support for Hardware Cursor: Movement of the mouse cursor significantly impacts the perceived end to end (E2E) latency of the Miracast session. E2E latency is how long it takes between a user moving the mouse and that movement being rendered on the Miracast display. Before hardware cursor, the mouse was a part of the encoded video

stream, having the same E2E latency as the Miracast stream. Typically, this was over 100 milliseconds. Hardware cursor adds the ability to send the mouse appearance and coordinates to the sink directly, to composite them over the Miracast stream. Sending the mouse data out of band from the Miracast streams allows the mouse latency to drop below 30 milliseconds.

Media features

Microsoft's Miracast RTSP stack is extended to provide extended functionality, better stream quality, better end-to-end latency, and useful information while diagnosing Miracast failures.

Diagnostic capabilities

The Microsoft Miracast stack implements a number of 1st and 3rd party extensions detailed in MS-WFDPE. Some of these extensions are useful while diagnosing Miracast failures and creating reports. These extensions include:

Support for Intel friendly name: Microsoft Miracast receiver supports Intel's extension for a friendly (human readable) receiver name. This helps to diagnose and report failures.

Support for Intel's receiver device URL: Microsoft Miracast receiver supports Intel's extension for providing a product information URL for the Miracast sink. This helps to diagnose and report failures.

Support for Intel's receiver manufacturer logo: Microsoft Miracast receiver supports Intel's extension for specifying an image representing the manufacturer of the Miracast sink. This helps to diagnose and report failures.

Support for Intel's receiver manufacturer name: Microsoft Miracast receiver supports Intel's extension for specifying the name of the manufacturer of the Miracast sink. This helps to diagnose and report failures.

Support for Intel's receiver model name: Microsoft Miracast receiver supports Intel's extension for specifying the model name of Miracast Sink. This helps to diagnose and report failures.

Support for Intel's receiver version: Microsoft Miracast receiver supports Intel's extension for specifying the firmware of the Miracast sink. This helps to diagnose and report failures.

Support for enhanced diagnostics: Microsoft Miracast source supports an enhanced diagnostics protocol extension that enables the Miracast Receiver to report error and reason codes to the Miracast Source.

Support for source receiver version: Microsoft Miracast receiver supports display source identification, which allows the version number of the Miracast source and allows for a unique identifier for the current Miracast session. These attributes help to diagnose and report failures.

Support IDR requests: Microsoft Miracast source supports sending instantaneous decoder refresh (IDR) requests during an RTSP M13 message. These IDRs can improve the video stream if too many dropped packets have occurred or if the Miracast receiver cannot itself request an IDR frame.

Support for additional video formats: Microsoft Miracast source supports additional video formats, including:

- CEA resolutions and refresh rates with extension
- VESA resolutions and refresh rates with extension
- Display native resolution refresh rates with extension
- Profiles bitmap with extension
- Levels bitmap with extension
- 4K and HEVC encoding, when available

Performance enhancements

Microsoft supports extensions to make Miracast sinks more performant in certain situations. These extensions are:

End of frame marker: Microsoft Miracast source repurposes the M-Bit of the RTP packet header to denote the end of the frame in the RTP packet. A Miracast receiver looking for the M-bit can save time by starting to decode the frame contained in the RTP packets instead of waiting for the next RTP packet to arrive with the header of the next frame.

Latency management: There are always tradeoffs between providing higher stream quality and allowing lower latency. Interactive content benefits from having minimal latency and can compromise somewhat on quality. Conversely, media content is best when there is a high-quality stream; compromises on latency are acceptable. Microsoft enables the receiver to dynamically react to what the user is doing. By introducing latency roles, a Miracast receiver can adjust its latency based on the user's intent.

Value	Description

Value	Description
Low	Specifies that the Wi-Fi Display Receiver SHOULD keep latency under 50 milliseconds.
Normal	Specifies that the Wi-Fi Display Receiver SHOULD keep latency under 100 milliseconds.
High	Specifies that the Wi-Fi Display Receiver SHOULD buffer additional frames to ensure smooth playback, as long as the latency stays under 500 milliseconds.

Stream enhancements

Microsoft Miracast source supports extensions that produce better stream quality. These extensions are:

Support for CABAC: Microsoft Miracast source adds support for Context-Adaptive Binary Arithmetic Coding (CABAC) --for AVC (H264) Baseline Profile and High Profile. This produces more efficient encoding and therefore uses less bandwidth to transmit the Miracast stream.

Dynamically switching the resolution and frame rate The dynamic resolution and refresh rate extension allows the Miracast Source to change the video resolution or video refresh rate of the video stream without sending an additional Real-Time Streaming Protocol (RTSP) message to the Miracast Sink. If supported, the Miracast receiver monitors the sequence parameter set/picture parameter set (SPS/PPS) in the H.264 stream for changes to the video resolution or video frame rate. The receiver adapts to such changes without displaying any visible changes such as flicker or a black screen.

Extended functionality

Microsoft supports extensions to make Miracast sinks more performant in certain situations. These extensions are:

UIBC Support: User Input Back Channel (UIBC) is an optional feature for sending input commands from the Miracast receiver to the Miracast sender for processing. The user can then interact naturally with the Miracast receiver's peripherals. For example, Keyboard/Mouse/Touch/Pen inputs attached to a Miracast receiver can act as input to the Miracast Source.

Microsoft real time bitrate modulation: The Microsoft Miracast source supports RTCP to track a user's current network condition. Using packet information from RTCP in conjunction with format change capability, the Microsoft Miracast source modulates the bitrate to provide a smooth streaming experience even in poor network conditions. In

addition, if a user's network conditions are good, the bitrate increases, providing a better quality stream.

Wireless Projection over an existing Wi-Fi network

Since the release of the Creators Update, Windows has been updated to support Wireless Projection over an existing Wi-Fi network. It is implemented on all Windows devices running the Creators Update including Windows, Surface Hub, Surface Hub 2S and Xbox.

Windows Wireless Projection over Infrastructure functionality leverages and builds upon functionality originally provided by Miracast such as using Wi-Fi Direct for discovery and using RTSP for transporting the video stream but includes a separate mechanism to identify the route to the receiver over the existing network connection.

Receiver Manufacturer Checklist

Feature	OS Support	Reliance on Driver Support	Receiver Support
WPS Information Element attributes for Telemetry	Windows 10	N/A	Required
IP Address Allocation in EAPOL-key frames	Windows 10	N/A	Recommended
Extended Channel Switch Announcement	Windows 10	Wi-Fi Driver	Recommended
Wireless Projection over an existing Wi-Fi network	Windows 10 Creators Update	N/A	Strongly Recommended
Hardware Cursor	Windows 10	N/A	Recommended
Microsoft Real Time Bitrate Modulation, Support for Additional Video formats	Windows 10 Anniversary Update	N/A	Recommended

Feature	OS Support	Reliance on Driver Support	Receiver Support
Latency Switching, Dynamic Resolution/Frame Rate, End of Frame Marker, CABAC Support, UIBC Support, Dynamically switch the Resolution and Frame Rate; Support IDR requests; Support for Source Receiver Version; Support for Enhanced Diagnostics; Support for Intel's Receiver Version; Support for Intel's Receiver Model Name; Support for Intel's Receiver Manufacturer name; Support for Intel's receiver Manufacturer logo; Support for Intel's Receiver Device URL; Support for Intel Friendly name	Windows 10	N/A	Optional

Implementing Wireless Projection over Miracast

Article • 05/19/2021

Windows 10 has supported wireless projection over Miracast since its first release. Over time further enhancements and improvements have been made to the quality and ease of the projection experience.

We recommend that Miracast sink manufacturers follow the specification as it is defined in the Wi-Fi Display Technical Specification v2.1 and the guidelines provided throughout this document to implement similar sink side functionality.

Implementing Wireless Projection over an existing Wi-Fi network

Article • 10/16/2019

Since the release of the Creators Update, Windows has been updated to support Wireless Projection over an existing Wi-Fi network. It is implemented on all Windows devices running the Creators Update including Windows, Surface Hub, Surface Hub 2S and Xbox.

Windows Wireless Projection over an existing network functionality leverages and builds upon functionality originally provided by Miracast such as using Wi-Fi Direct for discovery and using RTSP for transporting the video stream but includes a separate mechanism to identify the route to the receiver over the existing network connection.

We recommend that Miracast sink manufacturers follow the specification as it is defined in MS-MICE and the guidelines provided in the following section to implement similar sink side functionality.

The Mandatory aspects of Wireless Projection over an existing Wi-Fi network

Over time the MS_MICE specification has grown and become more complex as additional functionality has been added. The next section of this documentation is designed to help an implementor understand the minimal baseline functionality needed to deliver the Wireless Projection over an existing network method on a receiver.

Only a small number of changes need to be made to your Receiver in order to support wireless projection over an existing network. They break down as follows:

1. Modify the Information Elements (IEs) that your receiver emits
2. Enable your mDNS responder to send the proper response
3. Implement a TCP listener on port 7250 and handle 2 specific messages

Modify the IEs in your Receiver

The first set of changes you should make to your receiver is to implement the Vendor extension attribute defined in section 2.2.3 of the MS-MICE spec. It is required that every

Beacon and Probe Response the Receiver sends MUST include the Vendor Extension attribute.

Within the vendor extension attribute you need to define two sub-attributes to enable basic functionality:

1. The Host Name Attribute
2. The Capability Attribute

Additionally, within the Capability Attribute only two values need to be set:

- A - MiracastOverInfrastructureSupport (1 bit): 0 = not supported, 1 = supported
- C - Version (3 bits): The version of this protocol, which is 0x1

All other bits can be set to 0

Responding to mDNS queries

The Miracast Receiver MUST register the following records with the Receiver's local mDNS implementation.

- An SRV record associated with port 7250
  - <instance name>._display._tcp.local
  - Where the <instance name> is the friendly name of the Receiver
- And the following TXT key-value pair
  - Key: container_id
  - Value: A GUID that identifies the Receiver.

Protocol Handling

The Receiver needs to perform two steps:

1. The Receiver MUST start listening on TCP port 7250 for an inbound connection.
2. The Receiver needs to be able to receive and respond to the minimal set of protocol messages defined in the next section.

While 6 Messages are defined in the MS-MICE specification, the Receiver only needs to receive and respond to the following 2 Messages in order to implement basic

functionality:

- SOURCE_READY
- STOP_PROJECTION

Messages	Section	Description
SOURCE_READY 0x01	[2.2.1]	Indicates the Miracast Source is ready to accept a connection on the RTSP port. Mandatory
STOP_PROJECTION 0x02	[2.2.2]	Indicates the end of the projection. Mandatory
SECURITY_HANDSHAKE 0x03	[2.2.3]	Used to exchange DTLS handshake messages to initiate a connection with encryption of the multimedia stream. Optional
SESSION_REQUEST 0x04	[2.2.4]	Indicates the Miracast Source intends to connect to the Sink using the specified options. Optional
PIN_CHALLENGE 0x05	[2.2.5]	Sent by the Miracast Source to initiate the session using the PIN displayed by the Miracast Sink. Optional
PIN_RESPONSE 0x06	[2.2.6]	Sent by the Miracast Sink in response to a PIN_CHALLENGE received from the Miracast Source. Optional

While 7 TLVs are defined in the MS-MICE specification, only the following 3 TLVs are required for basic functionality:

1. FRIENDLY_NAME
2. RTSP_PORT
3. SOURCE_ID

TLV	Section	Description
FRIENDLY_NAME 0x00	[2.2.7.1]	Specifies the friendly name of the Miracast Source. Mandatory
RTSP_PORT 0x02	[2.2.7.2]	Specifies the port on which the Source is listening for RTSP connections. Mandatory
SOURCE_ID 0x03	[2.2.7.3]	Specifies an identifier for the Source, which is used for all messages sent during a Miracast session. Mandatory
SECURITY_TOKEN 0x04	[2.2.7.4]	Contains a DTLS handshake message. Optional

TLV	Section	Description
SECURITY_OPTIONS 0x05	[2.2.7.5]	Specifies whether stream encryption and/or PIN entry will be used for the session. Optional
PIN_CHALLENGE 0x06	[2.2.7.6]	Contains a salted hash of the PIN when PIN entry is used to establish the connection. Optional
PIN_RESPONSE_REASON 0x07	[2.2.7.7]	Specifies whether the PIN Response indicates a successful connection. Optional

Where to find additional implementation details on each feature

Article • 11/05/2020

WPS Information Elements (IEs) for Telemetry - implementation details

We request that you fill in the corresponding WPS Information Element (IE) attributes in WPS. The specific mappings that we use for Windows are:

- **WPS:Manufacturer**: Please put in the name of the Manufacturer, likely OEM or OEM + Chipset IHV, ex: Microsoft
- **WPS:Model**: This is the Unique Model for the device, ex: MWDA
- **WPS:Model-Number**: This is used to store the Firmware Version, ex: 1.3.8350

You can validate that the Windows 10 device can properly parse the Miracast receiver's Manufacturer, Model and Model-number values by performing the following:

1. Connect to the Miracast receiver
2. Go to Device Manager and look under Software Devices
3. Locate the Miracast receiver name
4. Look at its Properties. On the Details tab look for the following 3 properties: PnP-X manufacturer, PnP-X model name, and PnP-X model number.

IP address allocation in EAPOL-key frames implementation details

The Wi-Fi_P2P_Technical_Specification_v1.7 specification contains new methods to do IP address allocation during connection establishment instead of performing post-connection DHCP. Section *4.2.8, IP Address Allocation in EAPOL-Key Frames (4-Way Handshake)* shows how to implement this functionality.

Extended channel switch announcement (eCSA) implementation details

Extended Channel Switch Announcement is defined in the 802.11-2012 specification. This specification covers the implementation in detail in several sections including 6.3.37, 8.5.8.7, 8.4.2.55, and 10.10, among others. Please refer to this specification for technical information.

Wireless Projection over an existing Wi-Fi network

With the Windows Creators Update (1703) support for wireless projection over Infrastructure has been added to the Window receiver. This includes, but is not limited to, all Windows devices running the Creators Update including Windows PCs, Surface Hub and Xbox. We recommend that Miracast sink manufacturers follow the guidelines that are defined in the [MS-MICE] document to implement similar sink side functionality.

Support for H/W Cursor implementation details

A Miracast sink vendor can use the hardware cursor validation tool and a small suite of cursors to validate their hardware cursor implementation. For additional details please see [MS-WDHCE]: Wi-Fi Display Protocol: Hardware Cursor Extension

Window 10 Miracast Extensions implementation details

Windows 10-based Miracast source and sinks support all the Miracast extensions listed above. When implementing any of these extensions you can validate that the extension is supported using an over-the-air sniffer. Section 3 of [MS-WFDPE]: Wi-Fi Display Protocol Extension provides an M3 request and response for each extension defined in this document.

Feature	OS Support	Documentation
WPS Information Element attributes for Telemetry	Windows 10	This document and the Wi-Fi_Simple_Configuration_Technical_Specification_v2.0.6
IP Address Allocation in EAPOL-key frames	Windows 10	Wi-Fi_P2P_Technical_Specification_v1.7

Feature	OS Support	Documentation
Extended Channel Switch Announcement	Windows 10	802.11-2012 Specification
Wireless Projection over and existing Wi-Fi network	Windows 10 Creators Update	MS-MICE
Hardware Cursor	Windows 10	MS-WDHCE
Microsoft Real Time Bitrate Modulation, Support for Additional Video formats	Windows 10 Anniversary Update	MS-WFDPE
Latency Switching, Dynamic Resolution/Frame Rate, End of Frame Marker, CABAC Support, UIBC Support, Dynamically switch the Resolution and Frame Rate; Support IDR requests; Support for Source Receiver Version; Support for Enhanced Diagnostics; Support for Intel's Receiver Version; Support for Intel's Receiver Model Name; Support for Intel's Receiver Manufacturer name; Support for Intel's receiver Manufacturer logo; Support for Intel's Receiver Device URL; Support for Intel Friendly name	Windows 10	MS-WFDPE

Validating your implementation

Article • 10/16/2019

Validation of each of the features described within this document and its associated texts is necessary in order to ensure that the solution you are developing will land in the hands of your customers with high quality. This section provides a set of recommendations on how to validate functionality and maximize quality.

There are 3 primary means of validation testing:

Expand table

Validation	Validation Method	Checklist
Wi-Fi Driver verification	Microsoft Hardware Lab Kit (HLK)	Required
Wi-Fi Interoperability Testing	Wi-Fi Alliance (WFA) Miracast testing	Strongly Recommended
Wireless Projection Stress Testing	Windows Wireless Projection Stress Lab and Testing	Strongly Recommended

Wi-Fi Alliance (WFA) testing for Miracast

If you are either a Wi-Fi device manufacturer or a Miracast Receiver manufacturer you will want to ensure that, even if you do not intend to obtain certification of your device, you at least pass the WFA Wi-Fi CERTIFIED Miracast Test Plan tests to ensure a base level of interoperability with other devices in the ecosystem.

Windows Hardware Lab Kit (HLK) Validation

If you are a Wi-Fi device manufacturer then the Windows driver that you build for your device MUST be successfully validated through the Windows Hardware Lab Kit (HLK) tests. To support Miracast your device MUST pass the following suites of tests:

- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicDiscoveryTests.*
- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicPairingTests.*
- Microsoft.Test.Networking.Wireless.WiFiDirect.BasicReconnectTests.*

Details on the Windows Hardware Lab Kit (HLK) can be found on the download page for the HLK.

Windows Stress automation

The testing that is performed above through the Windows HLK and WFA certification is necessary and foundational. These two methods of testing yield a guarantee that an implementation conforms to the protocols defined for Miracast. While this is necessary, it does not reflect the full interoperability complexity that is felt by users once they attempt to use Miracast in their own environments with their own hardware and software combinations.

In response to this, Windows has created a stress test environment that allows the automation of any Windows (PC) Source for the purpose of performing pairing and reconnection attempts to a given Miracast Receiver. The benefit of leveraging this stress setup and automation is that the automation will gather up any needed OS traces, over the air sniffer traces, and can be extended to include your own driver or firmware tracing. When a failure is captured using this automation all the collateral needed to diagnose the root cause of the failure is present in a single comprehensive package that is ready for debug analysis.

Partners who have been onboarded to the Microsoft Collaborate partner portal can download the Miracast stress tool package here:

Client | UTIL | Miracast | Miracast stress test automation toolkit

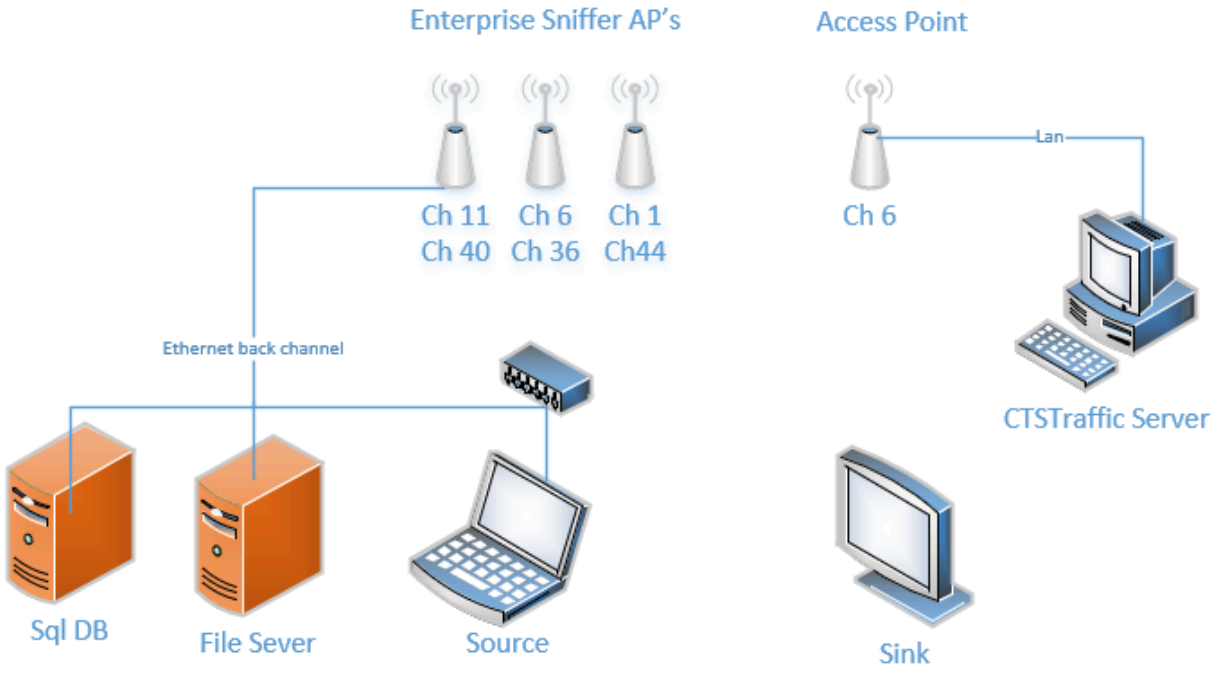
https://partner.microsoft.com/dashboard/collaborate/packages/7576

This package contains Miracast stress test tools and scripts to automate client / sink Miracast connection testing to help identify reliability issues.

Stress Lab Configuration and Materials list

Article • 05/19/2021

The full setup of the Windows Stress automation includes the elements shown in the diagram below:



In order to setup Widows Stress automation you will need the following devices and materials:

Expand table

Source (PC) machine	Identify the Windows PC to be used as the Source. You may use your own or pull from the IOP list provided in this document
	The Wi-FiAndWirelessProjectionStress_x64.msi file
	Xperf - installed via the Windows Assesment and Deployment kit (ADK)
Sink (Receiver) device	Use your own or pull from the IOP list
Sniffer devices	A Cisco 5500 wireless controller
	3 AIR-CAP3702E-A-K9 (configured for monitoring mode)
Server Machine	A Windows 10 machine (client or server will work for this)

	At least 2 additional hard drives (for storage of logs and over-the-air captures)
	Wireshark (for sniffer captures)
	7zip (for zipping the sniffer captures)
SqlDatabase	Any SQL database with the schema specified below. The SQL database is used to store the results of a test run.
Access Point	Any consumer Access Point will work

Configuring the Source (PC)

Partners who have been onboarded to the Microsoft Collaborate partner portal can download the Miracast stress tool package at:

Client | UTIL | Miracast | Miracast stress test automation toolkit

https://partner.microsoft.com/dashboard/collaborate/packages/7576

This package contains Miracast stress test tools and scripts to automate client / sink Miracast connection testing to help identify reliability issues.

1. Make sure that Windows PC is running the latest published OS version and drivers by going to Settings / Updates and Security and selecting Check for Updates.

2. After setting up your Wireless Projection Receiver verify that the client can connect to it.

3. Run Wi-FiAndWirelessProjectionStress_x64.msi

   a. Click **Next** on the welcome page.

   b. Select **Stress Client** on the install type, and then click **Next**.

   c. Select your optional settings:

   - "Disable Wireless projection over Infrastructure." – This will add the necessary reg key to disable Wireless projection over Infrastructure.
   - “Enable DriverVerifier” – This will enable driver verifier on the wifi adapter and the necessary support files. To get a list rules applied and files effected run verifier.exe /query from and elevated cmd prompt.

   d. Select the folders you wish to have the files copied to, and then click **Next**

> **Note**
>
> Wi-Fi stress is included in this package as well. Having it on the machine will not effect how the Wireless Projection stress is run.

e. Confirm the installation by clicking **Next**.

f. Confirm the unknown publisher request.

g. During the install a cmd window will open to complete some additional tasks.

- Enter in a new computer name if you wish to change it or click **Enter** to keep the current one.
- Type **y** or **n** to reboot the machine - if you enabled driver verifier or changed the name of your machine you will need to reboot before these take effect.

h. Click **Close** to complete the installation

4. Install xperf via the Windows Assessment and Deployment kit (ADK). /windows-hardware/get-started/adk-install
5. Connect wirelessly to the Infrastructure Access Point saving the profile as **Auto Connect**.
6. Validate all the necessary files were copied down:

Expand table

File	Description
c:\stress\Analyze_Miracast.exe	Utility that parses the ETL result files from the stress run and produces a result XML
c:\stress\Bucketizer.exe	Utility that takes in the XML output from Analyze_Miracast.exe, packages the various log files and handles the upload of the results to the SQL database for reporting.
c:\stress\Ctstraffic.exe	Traffic generating tool
c:\stress\ExecuteMiracastStress.ps1	Powershell execution script for running the stress

File	Description
c:\stress\P2papplication.exe	Utility for starting/stopping/querying Miracast connections
c:\stress\wpt\Perfcore.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\wpt\Perf_nt.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\Microsoft.Windows.EventTracing.GenericEvents.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\Microsoft.Windows.EventTracing.Interop.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\Microsoft.Windows.EventTracing.Processing.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\newtonsoft.json.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\Ionic.zip.dll	Used to compress results
c:\stress\System.IO.Compression.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\System.IO.Compression.FileSystem.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\System.Net.Http.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\System.Security.Principal.Windows.dll	Support DLL for Analyze_Miracast.exe for parsing ETL files
c:\stress\System.ValueTuple.dll	System.ValueTuple.dll - Support DLL for Analyze_Miracast.exe for parsing ETL files

File	Description
c:\stress\TraceProviders.txt	Minimum provider GUIDs for Miracast tracing
c:\stress\TraceProvidersReduced.txt	Full list of provider GUIDs for Miracast tracing

Configuring the Sink (Receiver)

Verify that the receiver you are using is running the latest version of the firmware. If you need a recommendation on a specific receiver to use please refer to our IOP matrix.

Configuring the Sniffers

1. On the controller make sure each radio in each AP is set to monitor mode and should be sending the traffic to the server machine.

   Make sure that you setup sniffing the following channels:

   - The social channels in the 2.4GHz band: 1, 6, and 11
   - The channel that the Access Point is on
   - The channel that the receiver is on

   In our example configuration we are using:

   - AP1 2.4 GHz = 1 (social channel), 5 GHz = 36 (access point)
   - AP2 2.4 GHz = 6 (social channel), 5 GHz = 40 (receiver)
   - AP3 2.4 GHz = 11 (social channel), 5 GHz = 44

2. On the server machine verify you are receiving the sniffer traffic from the controller (for reference see Collecting a Wireless sniffer trace using the Cisco Lightweight AP in Sniffer mode).

Configuring the server

1. Install Wireshark (note the installation directory)

2. Install 7zip (note the installation directory)

3. Enable Powershell Remoting

   i. From an elevated cmd prompt run:

```Powershell
> powershell
> set-netconnectionprofile -interfaceAlias \* -NetworkCategory Private
> enable-PSRemoting -Force -SkipNetworkProfileCheck
> set-Item wsman:\\localhost\\client\\trustedhosts \* -Force
> restart-service WinRM
```

4. Create two network shares:

   i. **Logs** - this is for copying the log files from the client ii. **Captures** - this is for storing the over the air captures that are collected on this server

5. Verify you can read/write to these shares from the client machine with credentials.

6. Note the username and password used to connect to these two shares and what the path to the shares is

SQL database:(optional)

1. Configure your SQL server with the specified schema

2. Once your SQL server is setup configure two tables with the schema's located in the DBSchema.txt file located in the zip under \ServerScripts

3. Make sure you can connect to your DB from the stress client as it will "sqlize" the results as the test runs

4. In order to sqlize the results you will need a dbconnectstring.txt file on your client in the stress directory. The following is an example of what should go in the dbconnectstring.txt file:

```SQL
Data Source=\<serverName\>;Initial Catalog=\<DBName\>;User
id=\<userid with write access\>admin
```

Configure the Access Point

1. Configure the Access Point with the following settings:

   a. SSID - <anything easy to find> b. Security Type: WPA2PSK AES c. Password: <something easy to remember>

Configure ctstraffic server

1. On the CtsTraffic server:

   a. Copy ctstraffic.exe to this machine b. From an elevated cmd prompt run:

   command

   ```
   ctsTraffic.exe -Listen:\* -Pattern:pushpull -PushBytes:4096 -
   PullBytes:4194304 -consoleverbosity:1
   ```

2. Plug this machine into the access point.

Configure Source side Sink Tracing (Optional)

If your sink supports debug output via a serial connection the stress script can collect this information and save it as part of the output

1. Install putty on the source (available on the web)
2. Connect the sink's debug output to the client
3. Validate output:

   a. Launch putty b. Open a serial connection using the com port the sink is connect to c. Verify out see the debug output d. Close putty

4. Note the COM port and baud rate used to connect to the sink as this will be passed into the stress script via parameters

Stress Testing Execution

Article • 06/26/2020

1. You will need the following items before staring your first run:

   a. The Machine Name of the Logs share

   b. Username and Password for the Logs share

   c. Full path to the Logs share

   d. The Sniffer capture machine name

   e. Username and Password for the Sniffer capture machine

   f. Full path to the Sniffer share

   g. dbConnectFile – if you plan on pushing your results to a SQL data base you will need to create a connection string to access that database. You can save it to a file and put it in the same location as your stress files

   h. Optional - com port and baud rate settings for putty if collecting sink side logging

   Example connect string:

```SQL
        Data Source=<sqldatabaseServer>;Initial
        Catalog=<sqldatabase;User id=<userid with write permissions>
```

2. Either modify the ExecuteMiracast.ps1 file with the above information or you can pass it in from an elevated cmd prompt. For example:

```command
       C:\stress> powershell.exe -file ExecuteMiracastStress.ps1 -
LogsMachineName "<logsMachineName>" -LogsMachineUserName "
<logsmachinename\useraccount>" -LogsMachineP "<logsmachine user password>" -
LogsShare "<\\<logsMachinename>\<LogsShare>" -dbConnectFile
"dbconnectstring.txt" -DoSniffCapture "true" -SnifferMachineName "
<SnifferCaptureMachineName>" -SnifferUsername "<snifferCaptureMachineName\
<useraccount>"  -SnifferP "<snifferuser's password>" -SnifferShare "<\\
<snifferMachine>\<captureShare>" -ComPortToUse = "<COM#>" -
PuttyConnectString:"-serial \\.\COM# -sercfg 115200,8,n,1,n"
```

3. Start stress:

   a. Based on step 2, either use the modified cmd line or if you modified the ExecuteMiracast.ps1 file directly you can just run `powershell -file ExecuteMiracast.ps1`

   b. When prompted enter in a runid – this number is used to track the run Note: this must be a number without spaces

   c. When prompted enter in a short description – this description is used to track the run

4. Stress should start

   Default execution time is 600 iterations or 18 hours whichever comes first. If you would like to change this just modify the "TotalIterations" or “MaxHoursToRun” parameters in the ExecuteMiracast.ps1 file.

Stress flow

1. Start CTSTraffic.exe to saturate the WLAN STA connection
2. Start logging

   a. Start local trace logging (Trace providers reduced, Trace providers full)

   b. Start sniffer capture on server

   c. Start xperf trace

   d. Start local ndis packet capture

3. Discover sink
4. First time pair to sink
5. Verify connectivity to sink
6. Disconnect from sink
7. Re-discover sink
8. Reconnect to sink
9. Verify connectivity to sink
10. Stop logging

a. Stop all local tracing

b. Stop OTA packet capture on server and zip it up

c. Analyze reduced xml file for any Miracast failures

d. Bucketize results*

- If any failures create a zip file of the following:
  a. Reduced .etl
  b. Full .etl
  c. Xperf .etl
  d. Local packet capture .etl
  e. Analyzed results .xml
  f. Build information

- If dbconnectstring.txt is present upload run results to a database named StressResults and StressDiscoveryFailures

e. Copy any bug zip files to server

f. Copy small etl and analyzed result xml to server

11. Repeat steps 2 - 10 for either totalIterations(600) or MaxHoursToRun (18) whichever comes first

> *The default parameters for the bucketizer will look at the result xml of the previous run to determine if there was a problem. The reason for this is that in the event of a problem the generated bug zip file will contain logs from the session before, during and after the found issue.

Miracast Stress Lab Result Interpretation

Article • 03/22/2021

Once the Miracast run is complete, you will notice that there are buckets of all failures created under the logs folder for the run. Please navigate to the right stress run iteration.

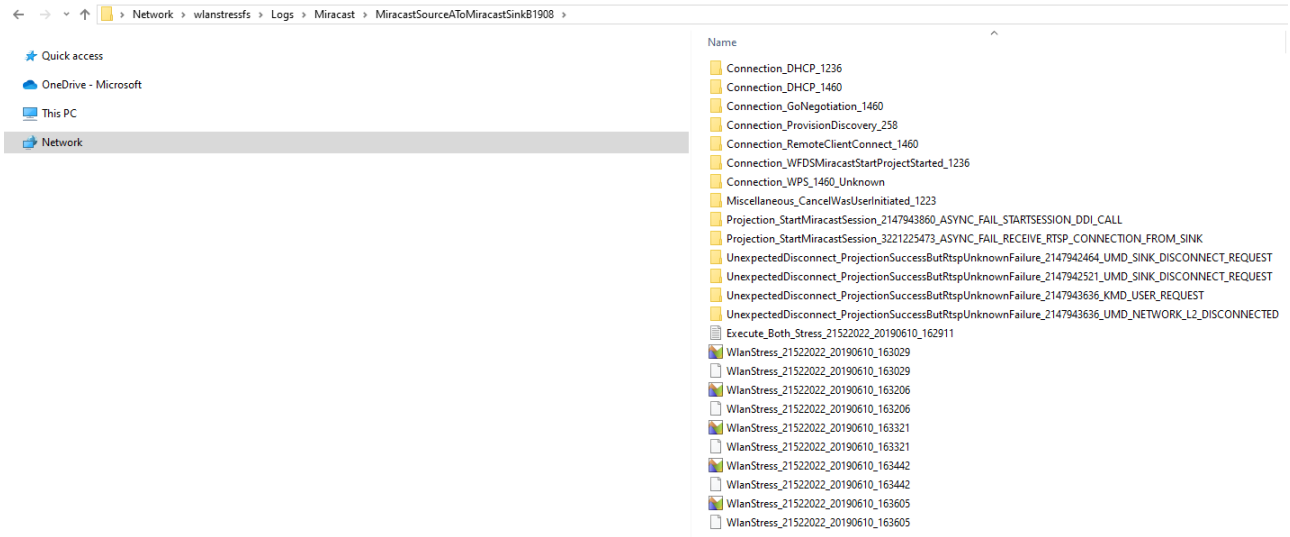
For example here are the bug folders created for a Miracast run:

Task ID: 22665191

Short Description: MiracastSourceAToMiracastSinkB1908

Run Start Time: 20190801_162606

Logs folder: \wlanstressfs\Logs\Miracast\22665191_MiracastSourceAToMiracastSinkB1908\20190801_162606



Unique failure bucket

Each folder here is a bug folder for each unique failure. Every unique failure bucket consists of three parts: Stage + State + ErrorCode. Please refer to the 4 tables below for the definitions for each Stage, State and common error codes. There is also a table that shows the correlation of every Stage and State to the expected over the air packets between the source and the sink. All blanks there indicate internal OS states.

What each failure bucket folder contains

Each failure bucket folder contains all the hits for that unique failure and the corresponding ETW OS traces and driver traces. We also include a before and after session for each failure. This has been super useful in some instances of bug investigations where a previous session's failure impacted the next session.

Session final stage definitions

#	Stage Name	Definition

#	Stage Name	Definition
1	Unknown	Unknown Begin or Unknown End: These are split sessions. We only get the session start telemetry or the session end telemetry due to 5pm UTC time split. Ignore/filter these sessions if it's just a few instances. If a lot of sessions fail this way, it might be a data bug. Report it as a bug with relevant logs in that case.
2	ProjectionSuccessButUnknownEnd	We get to first frame but the RTSP event and/or the stop Miracast event go missing. Ignore/filter these sessions if it's just a few instances. If a lot of sessions fail this way, it might be a data bug. Report it as a bug with relevant logs in that case.
3	Connection	Sessions that fail in establishing L2 connectivity: These are considered under "Get connected" failures
4	Projection	Sessions that fail in establishing the graphics/media setup : These are considered under "Get connected" failures
5	UnexpectedDisconnect	Sessions that were successful got connected (ie successful from user click to pixel/first frame was sent out to the sink) but dropped unexpectedly.
6	MissingRTSPEvent	Rtsp event goes missing for some sessions. This shouldn't happen as it is a critical event. But since this event has information whether the first frame was sent out or not, without that datapoint, we can't compute our get and stay connected metric. Ignore/filter these sessions if it's just a few instances. If a lot of sessions fail this way, it might be a data bug. Report it as a bug with relevant logs in that case.
7	UserInitiatedDisconnect	These are sessions which are successful from get and stay connected point of view. We successfully sent out the first frame to the sink and the disconnect was also initiated by the user.
8	Miscellaneous	UnknownPairing, ProjectionFailureButUserInitiatedDisconnect, Cancelled, InvalidWpsConfigMethod : These states should be ignored.

Session final state definitions

#	State Name	Definition
1	UnknownBegin	The beginning of the session is unknown.
2	UnknownEnd	The end of the session is unknown.
3	ShellCalledWFDSService	ConnectionStarted event from WFDS service
4	WFDSDafQuery	Initial query for data on AEP
5	WFDSDafQueryComplete	Initial query for data on AEP
6	WFDSConnectStarted	WFDS Connection complete event ConnectStarted state
7	InfraAttemptStarted	In case we do a connection over infrastructure (which fails) and Wi-Fi Direct
8	InfraChallengeIssued	Connection over Infrastructure challenge Issued
9	InfraChallengeComplete	Connection over Infrastructure challenge completed

#	State Name	Definition
10	WFDSWfdConnect	Wi-Fi Direct connect is for reconnects or Miracast "connect" after doing WSB pairing/connect. Invitation and KeyExchange happen here
11	DafRemoveAssociation	We remove association if it’s a forgotten pairing case or if the device is in bad state
12	DafRemoveAssociationComplete	Next State is : DafEnumCeremony : This is a Loop. Forgotten Pairing only happens during reconnects: wfdconnect (invitation or keyexchange) --> dafremoveassociation -> dafremoveassociationcomplete --> dafenumceremony
13	WFDSWfdConnectComplete	Wi-Fi Direct connect is for reconnects or Miracast "connect" after doing WSB pairing/connect (Invitation and Keyexchange should have completed by this stage)
14	EnumerateCeremonies	Begin session in WlanSvc. Enumerate Ceremonies encompasses the steps of the pairing process needed to determine which WPS ceremonies are available to be used for pairing. In most cases for Miracast, this will just be determined from the Probe Response frame, in which case no P2P Action frames are sent at this stage. In cases where we're inviting a device to connect to our local GO, this includes sending the invitation frame. The ceremony will be selected in this scenario when the remote device sends provision discovery.
15	SelectCeremonyComplete	This stage encompasses selecting a WPS ceremony. This occurs in software and doesn't involve any P2P action frame exchange. Failures at this stage usually indicate the connection requires PIN entry but is being triggered through a UX which doesn't support it.
16	Invitation	We'll begin the P2P Invitation action frame sequence here.
17	ProvisionDiscovery	Provision Discovery
18	ReadCeremony	
19	WriteCeremony	Waiting on user input. If write ceremony was without an error and the state was WaitingForPin, the user didn't enter a PIN. This step involves processing the PIN entry in the UX. It shouldn't involve any P2P action frame exchange.
20	PinEntered	PIN is entered by the user and something failed right after.
21	GoNegotiation	Go Negotiation

#	State Name	Definition
22	WPS	WPS Exchange. Please see the Wlan WPS Exchange Failure State field for details on where we failed. Send M1 to Send M8 is for the message exchange part. An "Unknown" state there refers to failures in association stage.
23	KeyExchange	KeyExchange
24	PairingUnknown	Something went wrong in pairing. Maps to DafFinalize
25	PairingComplete	Pairing complete. Also maps to DafFinalizeComplete
26	TargettedDiscovery	WlanSvc will perform a targeted discovery before pairing, if the driver's discovery cache doesn't contain a currrent discovery result.
27	StartGo	This is an internal state to Start GO: From wlansvc.connectioncomplete.
28	ConnectToRemoteGo	Connecting to the remote GO and completing the 4-Way Handshake : From wlansvc.connectioncomplete.
29	RemoteClientConnect	Waiting for a remote client to connect to our local GO and complete the 4-Way Handshake : From wlansvc.connectioncomplete.
30	IpAddressAssignment	From wlansvc.connectioncomplete
31	Association	Association state
32	WFDSWfdConnected	This is for Wi-Fi Direct First Pairing/WFD reconnect/Connection over Infrastructure. This is the end of WFDSConnectStarted.
33	Wi-FiDisplayConnectionStarted	"WirelessDisplay.ConnectionStart" event is fired : could repeat when infra fails and we fall back to Wi-Fi Direct.
34	Wi-FiDisplayConnectionComplete	"WirelessDisplay.ConnectionComplete" event is fired.
35	IpEndpointAndConfigureFirewall	Local firewall configuration after IP endpoints have been established.
36	DHCP	Retrieving the pair of IP endpoints for the session. Would be retrieved from WlanSvc and not involve any new protocol messages in the case of Fast-IP (aka. IP Address Assignment over EAPOL-Key frames).
37	Firewall	Firewall setup. Please note that firewall setup continues all the way till the RTSP port is setup. We won't Directly see failures due to firewall in the firewall bucket as the API calls are still made in the background.

#	State Name	Definition
38	OpenMiracastSessionComplete	"WirelessDisplay.OpenMiracastSessionComplete" : could repeat when infra fails and we fall back to Wi-Fi Direct. Low wlan events and also maps to WFDSMiracastConnectStarted.
39	WFDSMiracastConnectStarted	State right before we call into graphics driver start Miracast.
40	WFDSMiracastConnectComplete	The following states are expected to have completed: DHCP (could happen sooner) , Firewall, IP Endpoint and Configure Firewall, RTSP
41	InfraAttemptCompleted	Connection over Infrastructure Attempt completed and something failed right after it.
42	WFDSSinkPausing	When Projection to PC setting is enabled on the device, an outgoing Miracast connection from this source device can fail in sink pausing stage. This setup requires the device to pause the sink receiver state in order to enable an outgoing Miracast session. Typically this failure can disappear if the Projection to this PC setting is disabled or the connect app is closed and is no longer running in the background.
43	WFDSSinkPauseComplete	Wfds Connection manager is waiting for Miracast dev node to arrive.
44	WFDSWaitingForMiracastDevNode	This step involves waiting for the media-layer Miracast driver to intitialize itself and register a PNP device interface against the Wi-Fi Direct dev node. Failures at this step usually indicate an internal driver initialization issue in the OS.
45	WFDSMiracastDevNodeArrived	Miracast dev node has arrived and something failed right after it.
46	WFDSInfraBackChannelStarted	Wfds connection manager has started Connection over Infrastructure Back Channel
47	WFDSInfraBackChannelInitialComplete	Connection over Infrastructure Back Channel Initialization completed and something failed right after it.
48	WFDSInfraBackChannelFinalize	Connection over Infrastructure Back Channel Finalize started.
49	WFDSInfraBackChannelFinalizeComplete	Connection over Infrastructure Back Channel Finalize completed and something failed right after it.
50	WfdConnectDafQuery	During Wi-Fi Direct Connect, if we fail but have time to retry, we will trigger a DAF query before the next retry.
51	WfdConnectDafQueryComplete	Daf Query completed and something failed right after it.

#	State Name	Definition
52	WaitForBtScanningYielded	Before a Wi-Fi Direct Connection, we signal Bluetooth to yield scanning, and wait for them to signal they've yielded.
53	WaitForBtScanningYieldedComplete	Waiting for bluetooth scanning yield completed and something failed right after it.
54	WFDSMiracastStartProjectStarted	Indicates we've signaled the graphics stack to begin the projection stage. Start Miracast Session event should fire next.
55	WFDSMiracastStartProjectComplete	Projection completed- graphics and media set up is completed.
56	WFDSSessionConnected	End of entire setup phase of getting L2 connected and sending the first frame out.
57	WFDSUnknownFailure	WFDSUnknownFailure
58	UnknownPairing	Something failed in first time pairing, but the exact cause is unknown. You will see this bucket a lot for RS3 and older releases.
59	InvalidWpsConfigMethod	InvalidWpsConfigMethod
60	UnexpectedDisassociation	UnexpectedDisassociation
61	CancelWasUserInitiated	Cancel was apparently initiated by the user. In a stress environment, if there isn't enough time between the start of the session and when the disconnect call is issued, then it will appear as a user cancel as the session wasn't yet established. If this wasn't the case, then it could be some weird UI path that is initiating a cancel if this cancel wasn't actually expected at all.
62	StartMiracastSession	We called into the graphics stack for graphics driver setup and something failed.
63	StartMiracastCancelled	StartMiracastCancelled from the graphics layer
64	RtspMiracastM1	Something failed at RTSP M1 message exchange.
65	RtspMiracastM2	Something failed at RTSP M2 message exchange.
66	RtspMiracastM3	Something failed at RTSP M3 message exchange.
67	RtspMiracastM4	Something failed at RTSP M4 message exchange.
68	RtspMiracastM5	Something failed at RTSP M5 message exchange.
69	RtspMiracastM6	Something failed at RTSP M6 message exchange.
70	RtspMiracastM7	Something failed at RTSP M7 message exchange.
71	RtspMiracastStreaming	RtspMiracastStreaming failure
72	RtspFirstFrameFailure	RtspFirstFrameFailure - something failed right around the time the first frame was being sent out.
73	RtspUnknownFailure	RtspUnknownFailure

#	State Name	Definition
74	UserInitiatedDisconnect	For RS3 and older telemetry: Outcome is successful and the disconnect was user initiated.
75	UserInitiatedDisconnectAssumed	When we only get critical events we can only assume that the user initiated disconnect.
76	ProjectionSuccessAndUserInitiatedDisconnect	User clicked on the UI disconnect button after projection was successful
77	ProjectionFailureButUserInitiatedDisconnect	We didn't get to projection because the user clicked on the UI disconnect.
78	StopMiracastSession	For RS3 and older telemetry: A graphics stop event fires at the end of the session. StopSessionStatus of 0 indicates a successful and clean stop. There are several stop session reason codes that indicate an unexpected disconnect and those reason codes define some of the unexpected disconnect failure states.
79	AbnormalStopSession	For RS3 and older telemetry: Graphics stop event reported an abnormal disconnect. Please check the corresponding stop session reason code and the session final error code to know more about this session.
80	Teardown	For RS3 and older telemetry: Graphics stop event reported a Teardown. Please check the corresponding stop session reason code and the session final error code to know more about this session.
81	ProjectionSuccessButRtspUnknownFailure	Abnormal Disconnects due to some Rtsp Unknown Failure.
82	ProjectionSuccessButStopMiracastSession	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported a failure. Check the stop session reason code and the session final error code for more details for this session.
83	ProjectionSuccessButAbnormalStopSession	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an abnormal failure. Check the stop session reason code and the session final error code for more details for this session.
84	ProjectionSuccessButTeardown	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported a graphics teardown failure. Check the stop session reason code and the session final error code for more details for this session.

#	State Name	Definition
85	ProjectionSuccessButDisconnected_UnknownFailure	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an unknown failure. Check the stop session reason code and the session final error code for more details for this session.
86	ProjectionSuccessButDisconnected_UmdCriticalError	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode critical failure. Check the stop session reason code and the session final error code for more details for this session.
87	ProjectionSuccessButDisconnected_UmdMissingPackage	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode missing package failure. Check the stop session reason code and the session final error code for more details for this session.
88	ProjectionSuccessButDisconnected_UmdSinkDisconnectRequest	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode sink disconnect request failure. Check the stop session reason code and the session final error code for more details for this session. Also check sink side logs and sniffer logs.
89	ProjectionSuccessButDisconnected_UmdInsufficientBandwidth	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an insufficient bandwidth failure. Check the stop session reason code and the session final error code for more details for this session.
90	ProjectionSuccessButDisconnected_UmdSinkFailedStandardModeChange	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode SinkFailedStandardModeChange failure. Check the stop session reason code and the session final error code for more details for this session.
91	ProjectionSuccessButDisconnected_UmdNetworkL2Disconnected	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode L2 disconnect failure. Check the stop session reason code and the session final error code for more details for this session. Also check sink side logs and sniffer logs.
92	ProjectionSuccessButDisconnected_UmdPowerDownMiracastDevice	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode PowerDownMiracastDevice failure. Check the stop session reason code and the session final error code for more details for this session.

#	State Name	Definition
93	ProjectionSuccessButDisconnected_UmdWatchdogTimeout	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode WatchdogTimeout failure. Check the stop session reason code and the session final error code for more details for this session.
94	ProjectionSuccessButDisconnected_UmdCompanionDriverRequest	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an user mode CompanionDriverRequest failure. Check the stop session reason code and the session final error code for more details for this session.
95	ProjectionSuccessButDisconnected_KmdUserRequest	Reason for the stop Miracast session from kernel mode
96	ProjectionSuccessButDisconnected_KmdUserSessionSwitch	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode UserSessionSwitch failure. Check the stop session reason code and the session final error code for more details for this session.
97	ProjectionSuccessButDisconnected_KmdStopGraphicsDevice	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode StopGraphicsDevice failure. Check the stop session reason code and the session final error code for more details for this session.
98	ProjectionSuccessButDisconnected_KmdPowerDownGraphicsDevice	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode PowerDownGraphicsDevice failure. Check the stop session reason code and the session final error code for more details for this session.
99	ProjectionSuccessButDisconnected_KmdStopMiracastDevice	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode StopMiracastDevice failure. Check the stop session reason code and the session final error code for more details for this session.
100	ProjectionSuccessButDisconnected_KmdIsrInvalidData	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode IsrInvalidData failure. Check the stop session reason code and the session final error code for more details for this session.
101	ProjectionSuccessButDisconnected_KmdNoActivePaths	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode NoActivePaths failure. Check the stop session reason code and the session final error code for more details for this session.

#	State Name	Definition
102	ProjectionSuccessButDisconnected_KmdRemoveMiracastDevice	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode RemoveMiracastDevice failure. Check the stop session reason code and the session final error code for more details for this session.
103	ProjectionSuccessButDisconnected_KmdEnteringConnectedStandby	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode EnteringConnectedStandby failure. Check the stop session reason code and the session final error code for more details for this session.
104	ProjectionSuccessButDisconnected_KmdConnectedStandbyBackup	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode ConnectedStandbyBackup failure. Check the stop session reason code and the session final error code for more details for this session.
105	ProjectionSuccessButDisconnected_KmdInDirectSwapchainTimeout	This is an unexpected disconnect state. First Frame Projection was Successful but the graphics stop Miracast session event reported an kernel mode InDirectSwapchainTimeout failure. Check the stop session reason code and the session final error code for more details for this session.
106	ProjectionSuccessButUnknownEnd	We get to first frame (ActivityLog fires) but the RTSP event and/or the stop Miracast event go missing

Stages - states - over the air packets relation

Id	Stage	State	OTA
1	Unknown	UnknownBegin	
2	Unknown	UnknownEnd	
3	Connection	ShellCalledWFDSService	
4	Connection	WFDSDafQuery	
5	Connection	WFDSDafQueryComplete	
6	Connection	WFDSConnectStarted	
7	Connection	InfraAttemptStarted	
8	Connection	InfraChallengeIssued	
9	Connection	InfraChallengeComplete	
10	Connection	WFDSWfdConnect	
11	Connection	DafRemoveAssociation	
12	Connection	DafRemoveAssociationComplete	

Id	Stage	State	OTA
13	Connection	WFDSWfdConnectComplete	
14	Connection	EnumerateCeremonies	targeted discovery, invitation
15	Connection	SelectCeremonyComplete	
16	Connection	Invitation	
17	Connection	ProvisionDiscovery	Provision Discovery
18	Connection	ReadCeremony	
19	Connection	WriteCeremony	
20	Connection	PinEntered	
21	Connection	GoNegotiation	Go Negotiation
22	Connection	WPS	WPS
23	Connection	KeyExchange	
24	Connection	PairingUnknown	
25	Connection	PairingComplete	
26	Connection	TargettedDiscovery	
27	Connection	StartGo	
28	Connection	ConnectToRemoteGo	association
29	Connection	RemoteClientConnect	
30	Connection	IpAddressAssignment	
31	Connection	Association	
32	Connection	WFDSWfdConnected	
33	Connection	Wi-FiDisplayConnectionStarted	
34	Connection	Wi-FiDisplayConnectionComplete	
35	Connection	IpEndpointAndConfigureFirewall	
36	Connection	DHCP	DHCP
37	Connection	Firewall	
38	Connection	OpenMiracastSessionComplete	
39	Connection	WFDSMiracastConnectStarted	
40	Connection	WFDSMiracastConnectComplete	
41	Connection	InfraAttemptCompleted	
42	Connection	WFDSSinkPausing	

Id	Stage	State	OTA
43	Connection	WFDSSinkPauseComplete	
44	Connection	WFDSWaitingForMiracastDevNode	
45	Connection	WFDSMiracastDevNodeArrived	
46	Connection	WFDSInfraBackChannelStarted	
47	Connection	WFDSInfraBackChannelInitialComplete	
48	Connection	WFDSInfraBackChannelFinalize	
49	Connection	WFDSInfraBackChannelFinalizeComplete	
50	Connection	WfdConnectDafQuery	
51	Connection	WfdConnectDafQueryComplete	
52	Connection	WaitForBtScanningYielded	
53	Connection	WaitForBtScanningYieldedComplete	
54	Connection	WFDSMiracastStartProjectStarted	
55	Connection	WFDSMiracastStartProjectComplete	
56	Connection	WFDSSessionConnected	
57	Connection	WFDSUnknownFailure	
58	Connection	UnknownPairing	
59	Connection	InvalidWpsConfigMethod	
60	Connection	UnexpectedDisassociation	
61	Miscellaneous	CancelWasUserInitiated	
62	Projection	StartMiracastSession	
63	Projection	StartMiracastCancelled	
64	Projection	RtspMiracastM1	RTSP M1 Message
65	Projection	RtspMiracastM2	RTSP M2 Message
66	Projection	RtspMiracastM3	RTSP M3 Message
67	Projection	RtspMiracastM4	RTSP M4 Message
68	Projection	RtspMiracastM5	RTSP M5 Message
69	Projection	RtspMiracastM6	RTSP M6 Message
70	Projection	RtspMiracastM7	RTSP M7 Message

Id	Stage	State	OTA
71	Projection	RtspMiracastStreaming	
72	Projection	RtspFirstFrameFailure	
73	Projection	RtspUnknownFailure	
74	UserInitiatedDisconnect	UserInitiatedDisconnect	
75	UserInitiatedDisconnect	UserInitiatedDisconnectAssumed	
76	UserInitiatedDisconnect	ProjectionSuccessAndUserInitiatedDisconnect	
77	Miscellaneous	ProjectionFailureButUserInitiatedDisconnect	
78	UnexpectedDisconnect	StopMiracastSession	
79	UnexpectedDisconnect	AbnormalStopSession	
80	UnexpectedDisconnect	Teardown	
81	UnexpectedDisconnect	ProjectionSuccessButRtspUnknownFailure	
82	UnexpectedDisconnect	ProjectionSuccessButStopMiracastSession	
83	UnexpectedDisconnect	ProjectionSuccessButAbnormalStopSession	
84	UnexpectedDisconnect	ProjectionSuccessButTeardown	
85	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UnknownFailure	
86	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdCriticalError	
87	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdMissingPackage	
88	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdSinkDisconnectRequest	
89	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdInsufficientBandwidth	
90	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdSinkFailedStandardModeChange	
91	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdNetworkL2Disconnected	
92	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdPowerDownMiracastDevice	
93	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdWatchdogTimeout	
94	UnexpectedDisconnect	ProjectionSuccessButDisconnected_UmdCompanionDriverRequest	
95	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdUserRequest	
96	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdUserSessionSwitch	
97	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdStopGraphicsDevice	
98	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdPowerDownGraphicsDevice	
99	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdStopMiracastDevice	
100	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdIsrInvalidData	
101	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdNoActivePaths	
102	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdRemoveMiracastDevice	

Id	Stage	State	OTA
103	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdEnteringConnectedStandby	
104	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdConnectedStandbyBackup	
105	UnexpectedDisconnect	ProjectionSuccessButDisconnected_KmdInDirectSwapchainTimeout	
106	UnexpectedDisconnect	ProjectionSuccessButUnknownEnd	

Common error codes

Code	Definition	Error Description
1460	ERROR_TIMEOUT	This operation returned because the timeout period expired.
1236	ERROR_CONNECTION_ABORTED	The network connection was aborted by the local system.
31	ERROR_GEN_FAILURE	A device attached to the system is not functioning.
50	ERROR_NOT_SUPPORTED	The request is not supported.
5023	ERROR_INVALID_STATE	The group or resource is not in the correct state to perform the requested operation.
2404	ERROR_DEVICE_IN_USE	The device is in use by an active process and cannot be disconnected.
258	WAIT_TIMEOUT	The wait operation timed out.
648	ERROR_DEVICE_ENUMERATION_ERROR	The "%hs" encountered an error while applying power or reading the device configuration. This may be caused by a failure of your hardware or by a poor connection.
170	ERROR_BUSY	The requested resource is in use.
1225	ERROR_CONNECTION_REFUSED	The remote computer refused the network connection.
1223	ERROR_CANCELLED	The operation was canceled by the user.
2147500037	E_FAIL	
2147942414	E_OUTOFMEMORY	Ran out of memory
3221225473	STATUS_UNSUCCESSFUL	
3222093445	MF_E_SHUTDOWN	
2147942464	E_NETNAMEDELETED	The specified network name is no longer available.
3222094442	MF_E_NET_READ	Error reading from the network.
2147500033	E_NOTIMPL	Not implemented

Interoperability Testing Matrix

Article • 10/19/2020

With such a rich and diverse set of Miracast capable devices on the market it is important that testing involve a representative subset of this diversity to gain the best coverage of what Users will have in their environments. Any vendor can create their own interoperablity matrix, but we provide the following matrix to get the process started or to augment existing test matrices.

It is important to note that all the firmware and driver versions listed below were the leading firmware and driver versions at the time that this information was published. These driver and firmware version should be considered minimum values. **It is strongly recommended that each testing lab perform checks for the latest available firmware and drivers before testing begins to ensure the best results.**

Recommended devices for Source validation:

(Last Updated: Summer 2020)

Device Name	Driver or Firmware version
Surface Pro 6, OR Surface Book 2, OR Surface Laptop	Running 2004 + patches; Marvell Wi-Fi driver 15.68.17018.116
Surface Pro 7, OR Surface Book 3, OR Surface Laptop 3	Running 2004 + patches; Intel Wi-Fi driver 21.40.1.3
Surface GO	Running 2004 + patches; Qualcomm Wi-Fi driver 12.0.0.722
Dell Latitude 7390 (i5-8250U)	Running 2004 + patches (Intel 9560 running Wi-Fi driver 20.70.4.2; Intel Graphics driver 25.20.100.6472)
Android phone (Samsung Galaxy S8)	Android version 9.0

Recommended devices for Sink validation:

(Last Updated: Summer 2020)

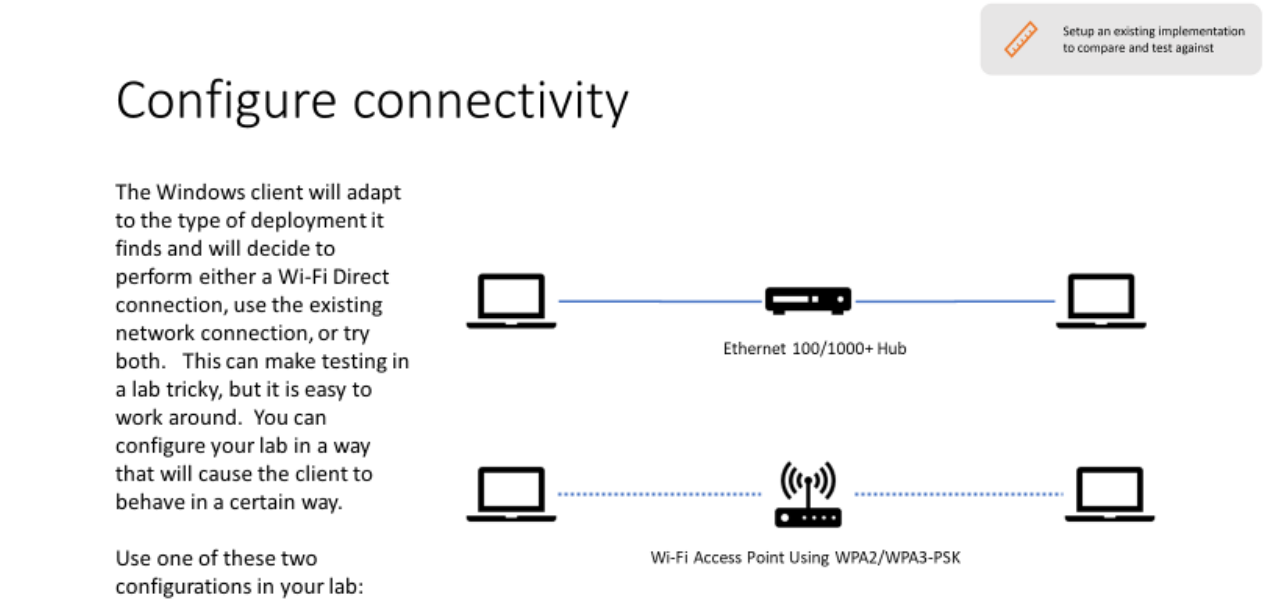
Device Name	Driver or Firmware version
Microsoft Wireless Display adapter v2	2.0.8384 (Wireless Display App available in the Microsoft App Store)

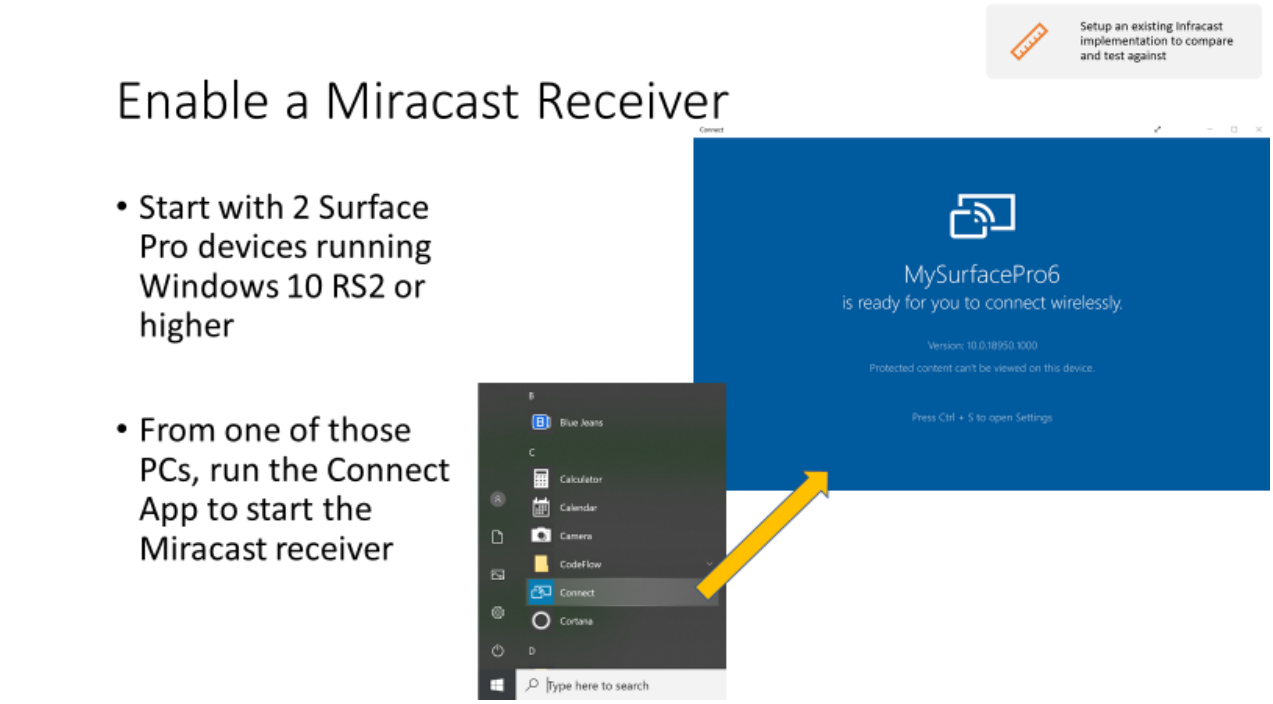
Device Name	Driver or Firmware version
Microsoft 4K Wireless Display Adapter	3.9520.30 (Wireless Display App available in the Microsoft App Store)
Surface Hub 2S	RS2 RTM + patches
Surface Studio	Running 2004 + patches
Xbox One S	2007 System Update
ScreenBeam 1100 Plus	11.1.8.0
ScreenBeam 1000	11.1.8.0
Actiontec Screenbeam Mini 2	5.3.15.0
LG TV (43UN7300PUF) [2020 model]	Running WebOS TV 5.0
Roku 3, OR Roku Premier	Firmware version 9.3
Samsung TV - (QN43Q60TAFXZA)	1180
Dell Latitude 7390 (i5-8250U)	Running 2004 + patches (Intel 9560 running Wi-Fi driver 20.70.4.2; Intel Graphics driver 25.20.100.6472)

For previous versions of our IOP recommendations please visit our IOP Archive.

Validating wireless projection over an existing Wi-Fi network to compare and test against

Article • 03/22/2021


Setup an existing implementation to compare and test against
Configure connectivity
The Windows client will adapt to the type of deployment it finds and will decide to perform either a Wi-Fi Direct connection, use the existing network connection, or try both. This can make testing in a lab tricky, but it is easy to work around. You can configure your lab in a way that will cause the client to behave in a certain way.
Use one of these two configurations in your lab:
Ethernet 100/1000+ Hub
Wi-Fi Access Point Using WPA2/WPA3-PSK


Setup an existing Infracast implementation to compare and test against
Enable a Miracast Receiver
• Start with 2 Surface Pro devices running Windows 10 RS2 or higher
• From one of those PCs, run the Connect App to start the Miracast receiver
MySurfacePro6
is ready for you to connect wirelessly.
Version: 10.0.18950.1000
Protected content can't be viewed on this device.
Press Ctrl + S to open Settings
Blue Jeans
Calculator
Calendar
Camera
CodeFlow
Connect
Cortana
Type here to search

Setup an existing Infracast implementation to compare and test against

Connect from the Miracast Source

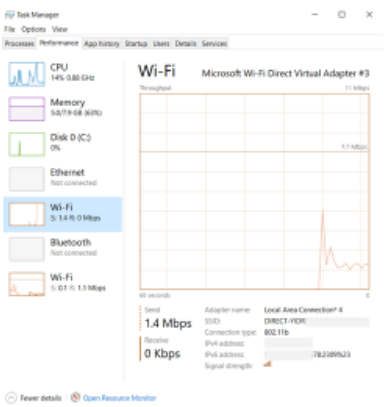
- On the second PC (the Miracast Source), Use the Win+K shortcut to pull up the list of nearby Miracast receivers
- Select the machine that you enabled as the Miracast Receiver and “Connect”

Verifying that a Miracast connection was established over Wi-Fi Direct vs. Infrastructure

When testing Miracast with the Windows Creators Update you may want to verify that a particular connection was established over either Wi-Fi Direct or the Infrastructure connection. Once you establish a Miracast session if you open the Performance tab in Task Manager you will either see a Wi-Fi connection with the term Direct in the SSID field (Figure **x**) or you will only see your Wi-Fi network connection (Figure **y**).

Setup an existing Infracast implementation to compare and test against

How to verify a Miracast connection was over either Wi-Fi Direct or Infrastructure



(Figure x)

From the Source device:

Once you establish a Miracast session if you open the Performance tab in Task Manager you will either see a Wi-Fi connection with the term “Direct-...” in the SSID field (Figure *x*) denoting a Wi-Fi Direct connection. Or you will only see your Wi-Fi network connection (Figure *y*) which implies an Infracast connection.

(Figure y)

Miscellaneous technical details and validation

Article • 03/18/2021

Verifying that you are running a WDI driver

You can validate that a Wi-Fi driver is a WDI driver by typing the following at a CMD prompt:

NETSH WLAN SHOW WIRELESSCAPABILITIES

This returns a list of values in the output including:

command

```
> WDI Version (Windows)
> WDI Version (IHV)
```

If these values are all zeros, (e.g. 0.0.0.0) then this is not a WDI driver. If these values are non-zero, a WDI driver is present. This is an example of the return values from a WDI driver:

command

```
> WDI Version (Windows) : 0.1.0.21
> WDI Version (IHV) : 0.1.0.1
```

Verifying that your PC can support Miracast

You can validate the Wi-Fi and Graphics drivers declare support for Miracast by typing the following at a CMD prompt:

`NETSH WLAN SHOW DRIVER`

This returns a list of values in the output including:

command

```
> Wireless Display Supported: Yes (Graphics Driver: Yes, Wi-Fi Driver: Yes)
```

Verifying that WPS Information Elements (IEs) are being sent properly

You can validate that the Windows 10 device can properly parse the Miracast receiver's Manufacturer, Model and Model-number values by performing the following: Connect to the Miracast receiver, then go to Device Manager, and look under Software Devices. Here you will see the Miracast receiver name listed. Look at its Properties. On the Details tab look for the following 3 properties: PnP-X manufacturer, PnP-X model name, and PnP-X model number.

Validating a receiver's properties without connecting

From an Administrator-level CMD prompt launch P2PApplication.exe and perform the following commands:

command

```
sccm

verbose

cmd
```

You will be presented with output such as the following which details properties from nearby receivers:

command

```
>        001) MSDisplayAdapter_91             : WiFiDirect#9A:5F:D3:03:0D:91
(9A:5F:D3:03:0D:91) Miracast
>                           InfraCast Supported: False
>                                  Manufacturer: Microsoft
>                                    Model Name: MWDA
>
>
>        002) jzen                            : WiFiDirect#DC:FB:48:00:54:1B
(DC:FB:48:00:54:1B) Miracast
>                           InfraCast Supported: True
>                                  Manufacturer: Microsoft
>                                    Model Name: Windows PC
>          InfraCast Stream Security Supported: True
>                       InfraCast PIN Supported: False
>                            InfraCast Hostname: jzen
>                      InfraCast IP Address(es): 10.137.188.135
```

```
>
2001:4898:d8:32:f5e5:2907:3a85:110f
```

Verifying that IP address allocation is occurring in EAPOL-key frames

To validate, use a wireless sniffer. The IP address request KDE and IP allocation KDE appear in the EAPOL-key exchange frames. Alternatively, there will also be an absence of the DHCP exchange.

Verifying that the PC's Wi-Fi driver supports Extended channel switch announcement (eCSA)

How to validate: You can validate that a WDI Wi-Fi driver supports eCSA by performing the following at a CMD prompt:

`NETSH WLAN SHOW WIRELESSCAPABILITIES`

You will see a list of values in the output including the following:

```
command

> Extended Channel Switch Announcement : Supported
```

Validating support for a hardware cursor

A Miracast sink vendor can use the hardware cursor validation tool and a small suite of cursors to validate their hardware cursor implementation. For additional info, see [MS-WDHCE]: Wi-Fi Display Protocol: Hardware Cursor Extension

Validating Window 10 Miracast extensions

Windows 10-based Miracast source and sink support all the above Miracast extensions. When creating a Windows 10-based Miracast source and sink you can validate that the extension is supported using an over the air sniffer. Section 3 of [MS-WFDPE]: Wi-Fi Display Protocol Extension provides an M3 request and response for each extension defined in this document.

Wireless Projection References

Article • 06/21/2022

- [MS-WFDPE]: Wi-Fi Display Protocol Extension
- [MS-MICE]: Miracast over Infrastructure Connection Establishment Protocol
- [MS-WDHCE]: Wi-Fi Display Protocol: Hardware Cursor Extension
- The 802.11-2012 specification (available through http://IEEE.org)
- Wi-Fi_P2P_Technical_Specification_v1.7 (available through http://www.wi-fi.org/)
- Wi-Fi_Simple_Configuration_Technical_Specification_v2.0.6 (available through http://www.wi-fi.org/)
- Wi-Fi Display Technical Specification v2.1 (available through http://www.wi-fi.org/)
- Details on the Windows Hardware Lab Kit (HLK) can be found in Windows Hardware Lab Kit.

Hardware component guidelines

Article • 06/08/2023

Building a great Windows device starts with selecting the right components. This section provides engineering guidance and recommendations for the following components.

In this section

Topic	Description
Audio	This document provides recommendations for the design and development of audio devices, including audio playback and audio input devices intended for use with Microsoft's Speech Platform.
Battery	This section provides battery performance recommendations.
Power Management for Storage Hardware Devices	This section provides power management guidelines for NVMe and SATA/AHCI storage devices.
Bluetooth	This topic provides a list of features and recommendations for Bluetooth in Windows 10. For power-efficient support for Bluetooth LE beacons and related scenarios, we recommend that OEMs use Bluetooth components capable of Hardware Offload and able to support the Microsoft-defined Host Controller Interface (HCI) extension.
Display	This topic covers guidelines for Display components in Windows 10.
Dynamic Lighting Devices	This topic covers guidelines for Dynamic Lighting devices in Windows 11.
Ethernet	This topic covers recommendations for Ethernet in Windows 10.
Graphics	This topic covers guidelines for Graphics in Windows 10.
Mobile Broadband	This topic covers recommendations for Mobile Broadband components in devices that run Windows 10.
Near Field Proximity (NFP/NFC)	This topic covers recommendations for near field communications (NFC) in Windows 10.

Topic	Description
Peripheral Component Interface Express	This topic provides recommendations for PCI Express (PCIe) in Windows 10.
Radial controller devices	This topic provides recommendations for radial controller devices in Windows 10.
Sensors	This topic provides recommendations for integrated sensors for Windows 10 and Windows 11.
Simple Peripheral Bus (SPB)	This topic covers recommendations for Simple Peripheral Bus in Windows 10.
Trusted Platform Module (TPM)	This topic provides recommendations for Trusted Platform Module (TPM) technology for Windows 10.
Windows Digitizer Class Input Devices	Touch interfaces can be found today across myriad devices ranging from mobile phones to slates, to kiosks, to horizontal/vertical displays of 30 inches. This section presents implementation and validation guides for the various Windows touch technologies.
Legacy Touchscreen and Pen Resources	The topics in this section present legacy resources for Windows Touchscreen and Pen devices. This information applies to Windows 8 and earlier operating systems.
Universal Serial Bus (USB)	This document provides recommendations for the design and development of USB components, including minimum hardware requirements, Windows Hardware Compatibility Program requirements, and other recommendations that build on those requirements. The goal of this document is to enable ecosystem partners to build a device with optimized USB functionality.

Audio

Article • 11/04/2022

This document provides recommendations for the design and development of audio devices, including audio playback and audio input devices intended for use with Microsoft's Speech Platform. The Speech Platform is used to power all of the speech experiences in Windows such as Voice Typing and Live Captions. The goal for this document is to enable ecosystem partners to build device with optimized audio experience with Microsoft technology.

Minimum hardware requirements and the Windows Hardware Compatibility Program

The minimum hardware requirements and Windows Hardware Compatibility Program requirements are fundamental for creating Windows-compatible audio solutions. Although the programs are optional, it is highly recommended that audio products meet both sets of requirements to ensure basic audio quality.

For more detail on each:

- See section 6.2.2 in Minimum hardware requirements
- See Windows Hardware Compatibility Program

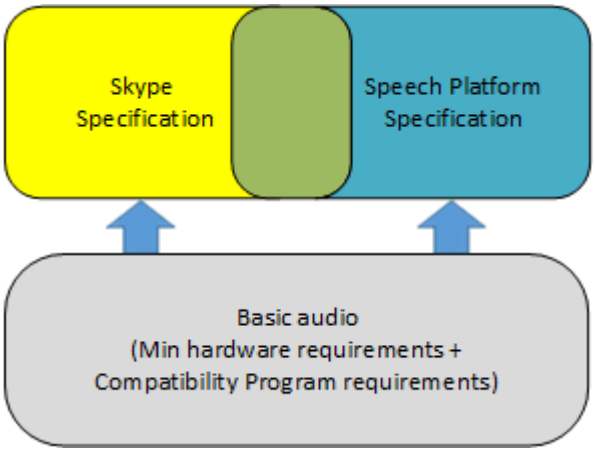
Scenario for optimizing multi-stream audio playback

Although Multi-streaming is no longer a requirement for the Windows 10 Desktop edition, it is highly recommended to have a minimum of two Digital-to-Analog Converters (DAC) to support Multi-streaming scenarios. If a single DAC (such as a Redirected headphone) is used, it is highly recommended to support audio volume control and status independently for each audio endpoint (such as e.g. integrated speakers, or a 3.5mm audio jack) so that user setting preferences can be retained.

Optimizing an audio solution for speech and communications

Once the audio solution meets both minimum hardware requirements and Windows Compatibility Program requirements, the audio solution will provide basic audio experiences in Windows. Depending on the targeted market segment, a device may support two additional optimizations: Speech Platform and Skype. Recommendations for both Speech Platform and Skype build upon the requirements for a basic audio experience. It would be a challenge to optimize for Speech Platform or Skype if the audio solution does not fully meet the basic requirements.

> ⓘ **Note**
>
> Guidelines for Telephony and applications such as Skype will be supplemented to this topic when they are available.



Speech recognition in Windows

Device manufacturers are encouraged to integrate and tune speech enhancement processing into their device in order to optimize performance relative to the Speech Recognition test criteria.

For devices without integrated speech enhancement processing, Microsoft provides default processing in Windows. The speech enhancement processing from Microsoft does not need device-specific tuning by the IHV.

The Microsoft speech enhancement pipeline will be used if the audio driver does not expose a microphone geometry and audio signal processing for speech. To utilize third-party enhancements, the microphone geometry must be provided, support “speech” signal processing mode on the audio input, and ensure that the effects provided by the audio driver or its software APOs include at least noise suppression and echo cancellation.

Difference between Speech Recognition and Telephony

Many devices that target Speech Recognition functionality will also target Telephony usage. The similarities are evident – both scenarios use devices with microphones to pick up human speech, audio processing pipelines to remove noise from the environment and enhance human speech, and consuming applications that rely on a clear speech signal in order to understand the speech.

The differences lie in who or what consumes the speech signal. Telephony has a human consumer, for whom perceptual voice quality and speech intelligibility are paramount. Speech recognition has an algorithmic consumer, where machine learning trained on specific features of the speech signal determine what is recognized, and where those features do not necessarily align to perceptual norms.

Perceptual voice quality is often correlated with speech recognition accuracy, but this is not always the case. This document focuses on methods of evaluating and maximizing speech recognition accuracy. It is recommended to support the “speech” signal processing mode and to tune that mode specifically for speech recognition.

Passing Skype/Lync audio certification is a strong indicator of good device audio performance.

Audio device recommendations

The following sections cover recommendations for speech. To ensure a high quality speech experience, all devices should be tested against these performance requirements.

Area	Type of guidance	Which devices should be tested
Device.SpeechRecognition	Provides the speech recognition performance requirements to ensure a high quality speech experience.	All devices should be tested against these performance requirements.
Device.Audio	Provides guidelines in order to function optimally with the host OS in terms of software interfaces, communication protocols, and data formats.	All devices should be tested against these guidelines.

Area	Type of guidance	Which devices should be tested
Device.Audio.Acoustics	Provides recommendations and best practices for acoustics and related properties of device design.	Most relevant for devices that will use Microsoft's speech enhancement processing.

Device.SpeechRecognition

The following table summarizes Microsoft recommendations for target speech recognition accuracy for devices in various environments. All targets are in speech accuracy.

Test	Description	Target	Recommendation
Device.SpeechRecognition.Quiet	An ideal environment with minimal ambient noise (noise floor < 35 dBA SPL) and no receive (echo path) noise.	Quiet <= 35 dBA SPL	95%
Device.SpeechRecognition.AmbientNoise	Various levels and types of noisy environments, e.g. Café & Pub.	Ambient Noise @ DUT >= 57 dBA SPL	90%
Device.SpeechRecognition.EchoNoise	Various levels and types of render playback scenarios (e.g. media playing).	Echo Noise @ LRP >= 70 dBA SPL	90%

Device.Audio

Recommendations in this section are made regarding the software and hardware interfaces, communication protocols, and data formats of the device. Devices intended to use speech recognition functionality must meet all Device.Audio Requirements.

Name	Recommendation
Device.Audio.Base.AudioProcessing	Drivers must expose all audio effect via the FXStreamCLSID, FXModeCLSID, and FXEndpointCLSID APOs (or proxy APOs). The APOs must send an accurate list of effects that are enabled to the system when queried. Drivers must support APO change notifications and only notify the system when an APO change has occurred.
Device.Audio.Base.StreamingFormats	Speech recognition functions under all audio capture and render streaming formats defined in the StreamingFormats HLK, with the optimum being 16 kHz with 24-bit capture and mono render.
Device.Audio.Base.SamplePositionAccuracy	It is important that render and capture audio signals are both 1) sampled accurately and 2) time-stamped accurately.
Device.Audio.USB.USB	All USB audio input devices must properly set descriptor according to the USB.org device class spec.
Driver Guidelines	Roadmap for developing WDM Audio Drivers

Device.Audio.Acoustics

Recommendations in this section are made regarding the acoustic and related properties of the device, such as microphone and loudspeaker placement, microphone responses, received noise from the device, and so on. Microphone selection, placement, integration, and array design are some of the most important factors to support quality speech recognition performance.

The recommendations and tests are relative to the signal before the speech enhancement processing but after microphone equalization and fixed microphone gain.

For more details about all of these recommendations, including recommended mic array geometries, see Microphone Array Geometry Descriptor Format.

Name	Recommendation

Name	Recommendation
Device.Audio.Acoustics.MicArray	Refer to Microphone Array Support in Windows. The audio driver must implement the KSPROPERTY_AUDIO_MIC_ARRAY_GEOMETRY property. Then the System.Devices.MicrophoneArray.Geometry property can be accessed via the Windows.Devices.Enumeration API. The USB audio driver will support this property for USB microphone arrays that have the appropriate fields set in the USB descriptor.
Microphone Array Descriptor	The device must describe its microphone type and geometry using the mic array descriptor.
Device.Audio.Acoustics.MicSensitivity	The Max recommendation is set to be able to support speech input levels considered to be “loud” and the Min recommendation is set to be able to support speech input levels considered to be “quiet.”
Device.Audio.Acoustics.MicIntegration	The microphones must be integrated to ensure a good acoustic seal between the microphone and the device chassis, and if appropriate, along the microphone porting tube. Minimize the acoustic noise and vibration between the system and the microphone. Two typical solutions are to use a rubber boot or a gasket. Whichever method is chosen check that the acoustic seal is sufficient across all production tolerances and over both environmental and lifetime changes.
Device.Audio.Acoustics.MicPlacement	Place the mic as far as possible from noise sources such as loudspeakers, fans, keyboards, hard drives, and the user’s hands, and as close as possible to the speaker’s mouth.
Device.Audio.Acoustics.MicSelfNoise	Use of a high quality microphone will minimize the microphone internal noise. Microphones with an SNR of at least 61 dB nominal is recommended for Standard and 63 dB for Premium.

Name	Recommendation
Device.Audio.Acoustics.MicReceivedNoise	The two main sources of received noise are acoustic noise and electrical noise. Acoustic noise may come from outside the device, or be generated internally in the device due to fans, hard disks etc. The acoustic noise can also be transmitted through the device mechanics. Electrical noise can be minimized by using digital microphones rather than analog microphones.
Device.Audio.Acoustics.MicMagnitudeResponse	The Premium and Standard masks apply to all device tiers under Device.Audio.Acoustics.Bandwidth e.g. a device can have Standard bandwidth (narrow-band) and a Premium magnitude response within that band.
Device.Audio.Acoustics.MicPhaseResponseMatching	This recommendation ensures that the temporal relationship between signals received via microphone elements in an array is consistent with the physical geometry of the microphone elements in the array.
Device.Audio.Acoustics.MicDistortion	The distortion is recommended to be measured using SDNR (pulsed noise signal-to-distortion-and-noise ratio), although THD targets are also given.
Device.Audio.Acoustics.MicBandwidth	The sample rate of the capture signal is the primary factor in determining the effective bandwidth of the speech signal. As the speech platform uses 16 kHz acoustic models in the speech recognizer, a 16 kHz minimum sample rate is recommended. 300 Hz is the effective lower end of the speech recognizer, however 200 Hz is the recommended acoustical limit for devices also targeting voice communications.
Device.Audio.Acoustics.RenderDistortion	The distortion is recommended to be measured using SDNR (pulsed noise signal-to-distortion-and-noise ratio), although THD targets are also given.

Name	Recommendation
Device.Audio.Acoustics.RenderPlacement	To enable the acoustic echo canceller to work well, the device speakers should be placed at a maximum distance from the microphones, or place directivity nulls towards loudspeakers.

Requirements to enable a third-party enhancement pipeline

The following requirements are key to enable a third-party enhancement pipeline. These and other recommendations are covered in more detail in the following sections:

- Microphone location reporting –explains how to implement a reporting structure for a mic array.
- Speech mode supports:
  - How to register APOs for specific modes
  - Audio signal processing modes
- Device.Audio.Base.Audioprocessing – Acoustic Echo Cancellation (AEC) and Nosie Suppression (NS) are required for third-party pipeline:
  - Implementing Audio Processing Objects
  - Audio Processing Object Architecture

Related resources

Windows Hardware Compatibility Program

Windows Hardware Lab Kit

Skype/Lync audio certification

Microphone Array Support in Windows

Battery

Article • 06/24/2024

For optimal battery performance review the following recommendations.

In this section

Expand table

Topic	Description
Battery and charging	This topic covers recommendations for battery and charging in Windows 10. All devices running Windows have a consistent battery charging experience, regardless of form factor, instruction set, or platform architecture. As a result, users have a consistent and quality experience with battery charging.
Battery Saver	The battery saver feature exists on Windows 11 24H1 (and earlier) and on Windows 10. It helps conserve power when a system is running on battery. When battery saver is on, some Windows features are disabled or behave differently.
Energy Saver	Energy saver replaces battery saver and exists on Windows 11 24H2 and later versions. It significantly extends battery life and reduces energy consumption by automatically managing power usage across your system and apps for an optimal balance of PC performance and energy efficiency.

Battery and charging

Article • 06/21/2022

Battery charging user experience

This topic covers recommendations for battery and charging in Windows 10. All devices running Windows have a consistent battery charging experience, regardless of form factor, instruction set, or platform architecture. As a result, users have a consistent and quality experience with battery charging.

1. Charging always occurs when connected to the charger.

   Except for cases of battery failure, a device running Windows is always capable of charging the battery whenever it is connected to the charger.

2. Windows can always boot when connected to the charger.

   - Windows 10 for desktop editions (Home, Pro, Enterprise, and Education):

     If the device is in the S5 (shutdown state), it can always boot into Windows when connected to the charger, regardless of the battery charge level and presence of the battery, if the battery is removable.

   - Windows 10 Mobile:

     A battery must be present and have enough charge level in order for the system to boot up.

3. Hardware autonomously manages charging.

   The hardware charges the device's battery without requiring firmware, Windows, drivers, or other software running on the main CPU(s). This requirement only applies to Windows 10 for desktop editions systems. Windows 10 Mobile systems may require the support of a UEFI charging app and/or other software components in order to charge the battery.

4. Charging stops automatically when the battery is fully charged or when a fault occurs.

   The hardware automatically stops charging when the battery is completely charged. This is done without requiring firmware, Windows, drivers, or other software running on the main CPU(s). If there is a battery or thermal fault condition, charging is also automatically stopped.

Charging occurs when connected to the charger

Users expect their device to charge whenever it is connected to the charger. As such, the hardware must always attempt to charge the battery whenever the device is connected to the charger regardless of the power state. This expectation holds true across all of the power states including active (S0), Sleep (S3), Hibernate (S4), shutdown (S5), hard off (G2/G3) and S0 Idle. Charging may stop once the battery is fully charged or if a fault condition occurs.

We do not recommend a design that charges the battery at a reduced rate when Windows or the firmware has not been booted or running. For example, the battery may charge at a slower rate when the system is completely off and connected to the charger and charge at a faster rate when the device is booted and ACPI firmware can be used to monitor the battery periodically.

Finally, a design may charge the battery at a lower rate when the system is in a thermal condition. In this scenario, heat may be reduced by slowing or eliminating battery charging altogether. Thermal conditions are the exception in any good system design.

Windows is always bootable when connected to AC power

- Windows 10 for desktop editions

  Users expect they can immediately boot and use their device whenever it is connected to the charger. As such, the device must always boot and be fully useable when connected to AC power. This holds true regardless of the battery charge level, battery/charger state, and battery presence (if the battery is removable).

  If the device requires a minimum battery capacity to boot the firmware and Windows, the hardware must ensure that battery capacity is always reserved by the platform. The reserved battery capacity must not be exposed to Windows.

- Windows 10 Mobile

  When the system is connected to AC power and the battery is present, the system should attempt to boot to the operating system, as long as the battery has sufficient charge to power the system during the boot process.

Hardware autonomously manages charging

As specified above, users expect their device to charge when it is connected to the charger. As a result, the hardware must charge the battery without requiring firmware, Windows, drivers or other software running on the main CPU(s) as one or more of these components may not be operational or may be in a fault state at any given time. This requirement applies only to Windows 10 for desktop editions systems. Windows 10 Mobile systems may require the support of a UEFI charging app and/or other software components in order to charge the battery.

Charging stops automatically when fully charged or when a fault occurs

The hardware automatically stops charging when the battery is completely charged or if a fault occurred. As with charging, this must be done without requiring firmware, Windows, drivers, or other software running on the main CPU(s). Further, the hardware is required to comply with all battery safety regulatory conditions.

Power and charging indicators

Windows provides a power source and battery status indicator using icons the user can see in several places. Places include the battery system tray icon and lock screen.

A device can also have a physical indicator such as an LED indicating the charging status. This indicator must have little impact on the power consumption.

Windows power and charging icons

Windows displays power source and charging status in three locations:

- On the lock screen:

  Windows displays a battery icon with the power source and charge status.

- Desktop system tray (Windows 10 for desktop editions only):

  Windows displays a battery icon with power source and charge status. When the user clicks on the battery icon, they can view information such as capacity remaining, estimated time remaining, and per-battery details (if equipped with multiple batteries).

- Status Bar (Mobile SKU only):

Windows displays a battery icon with power source and charge status. When the user swipes down from the top of the screen to expand the action center, they can view the actual battery percentage.

- Battery Saver Settings:

  In the Battery Saver settings page (Settings -> System -> Battery Saver), Windows displays the overall battery percentage, battery status (Charging vs. Discharging) and the Estimated Remaining Time to charge/discharge.

For platforms capable of S0 Idle, if the display is visible, Windows briefly lights up the display when the system is connected to or disconnected from the charger to notify the user of a power source change.

Platform hardware charging indicators

The icons built into Windows only address scenarios where Windows is running and the display is visible to the user. However, the on-screen indicators are not visible when the system is shutdown or S0 Idle state where the display is off. Because the user cannot see visual cues on the screen, the platform may include a physical charging indicator to indicate power is present.

The following section provides our recommendation for implementing keyboards and mice/touchpads on S0 Idle platforms with docking solutions. This section also talks about the challenges and principles along with potential solutions. Both potential solutions apply to mobile and A/C powered stationary docks.

Exposing the power and charging subsystem to Windows

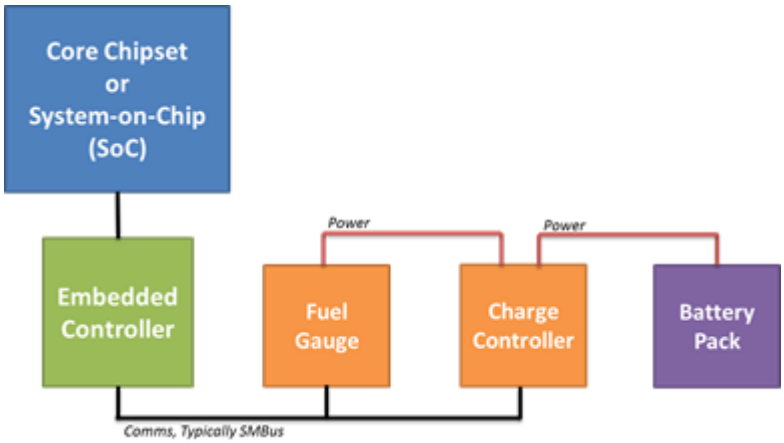
Every mobile device running Windows includes one or more batteries and a power source such as an AC adapter. Information from these subsystems conveys power management status to the user. The status includes remaining battery capacity at any time, state of the AC adapter and battery charging, and estimated remaining battery time. The power subsystem information is exposed in the Windows battery meter and other power management diagnostic utilities.

The following section provides our recommendation for implementing keyboards and mice/touchpads on S0 Idle platforms with docking solutions. This section also talks about the challenges and principles along with potential solutions. Both potential solutions apply to mobile and A/C powered stationary docks.

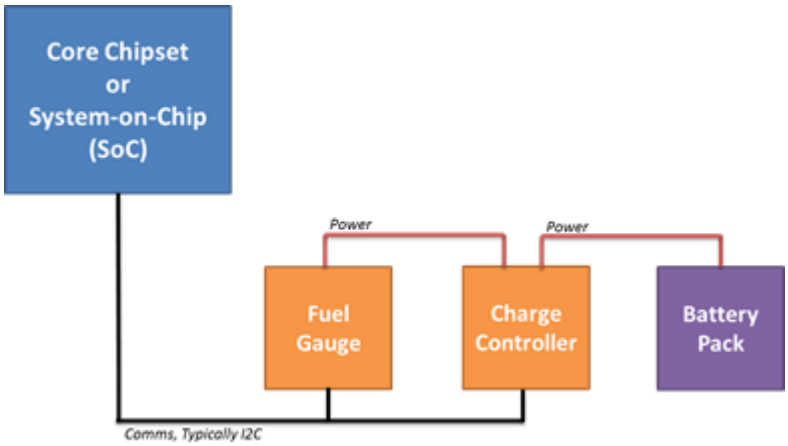
Typical power subsystem hardware topologies

Generally, Windows expects one of two hardware topologies for the power and charging subsystem.

The following figure illustrates the first topology that uses the platform's Embedded Controller, which is common in existing devices running Windows. The Embedded Controller performs multiple functions in a mobile device, including power source control, battery charge management, power button/switch detection and PS/2-compatible keyboard and mouse input. The embedded controller is typically connected to the core silicon through the Low Pin Count (LPC) bus. Windows queries and is notified about power subsystem information through the ACPI embedded controller interface.



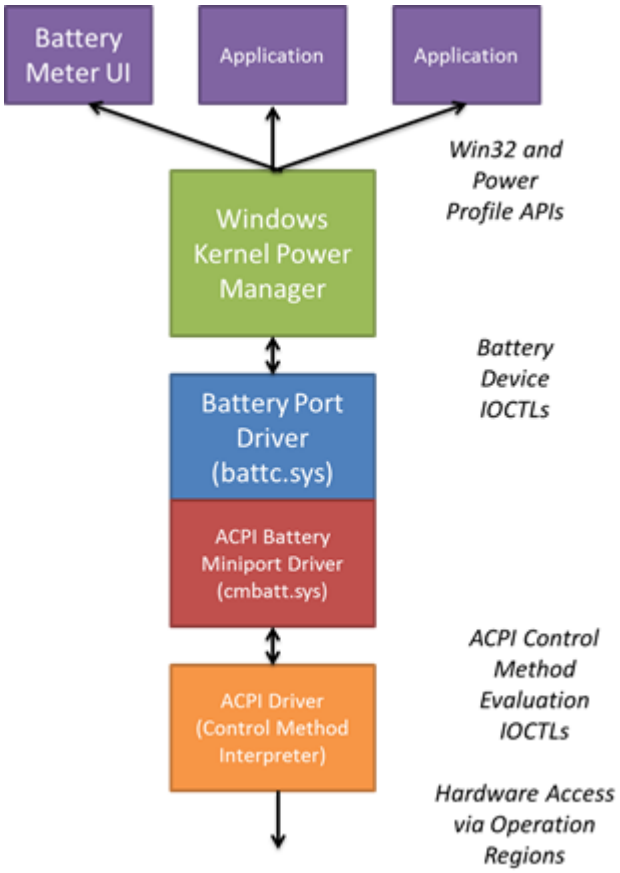
The next figure illustrates the second topology, which makes use of a battery charge controller and fuel gauge component connected directly to the platform's core silicon over a lightweight peripheral bus such as I^2C. In this configuration, Windows queries and is notified about power subsystem changes through communications over the I^2C bus. Instead of using a device driver for the battery or charging subsystem, the ACPI control method environment is extended with support for a Simple Peripheral (SPB) Operation Region. The SPB operation region allows the ACPI control method code to communicate to the battery charge controller and fuel gauge components connected to the core silicon over I^2C.



Battery and power subsystem driver model

Windows features a robust battery and power subsystem device driver model. The power management information is conveyed to the Windows power manager through a battery device driver, then aggregated and exposed to the Windows user interface through the battery device IRPs and a set of power management software APIs.

The battery driver model is a port/miniport model—that is, the battery model and interfaces are defined such that the new battery types can be exposed through a miniport. However, in practice, there are only two miniports that have any significant use in the Windows ecosystem - the battery miniport driver supporting the ACPI control method batteries and the HID battery miniport driver for USB-attached Uninterruptible Power Supply (UPS) devices.



All PCs are expected to expose the batteries and charging subsystem through the ACPI control method interface. The battery miniport interface should not be used for platform-specific battery charging subsystems. There are ACPI specification-defined control methods that allow Windows to poll for battery information and status. Similarly, there is an event-driven model to allow the hardware platform to notify Windows of battery and power source changes, such as a transition from AC to battery power.

Status polling

The Windows power manager periodically requests status information from the battery including the charge capacity remaining and the current rate of drain. This request originates in the power manager, a higher-level user interface component, or application. The power manager turns the request into an I/O Request Packet (IRP) to the battery device(s). When the battery is exposed through the ACPI control method interface, the control-method battery driver (cmbatt.sys) executes the appropriate ACPI control methods. In the case of status information, the _BST (battery status) method is executed.

The _BST method requires the ACPI firmware to obtain current information from the power subsystem and then packages that information in a buffer with format specified by the ACPI specification. The specific code required to access the battery status either from the embedded controller or the battery charger connected through I^2C is contained within the ACPI firmware and part of the code comprising the _BST method. The net result of the _BST method is the buffer of information required which is returned to the control-method battery driver. The control method battery driver finally converts the buffer to the format required by the battery driver and Windows power manager.

State change notifications

The power and battery subsystem will generate several notifications to Windows for state changes, including transitions from AC to battery power. Polling by Windows for these state changes is not practical given the high frequency at which polling would be required. Therefore, the hardware platform must use an event-driven model to notify Windows when the battery state changes significantly.

When the battery status changes, including remaining capacity or charging status, the ACPI firmware issues a Notify(0x80) on the control method battery device. The Windows control method battery driver then evaluates the _BST method and returns the updated information to the power manager.

When the battery static data changes, including last full charge capacity, design capacity and cycle count, the ACPI firmware issues a Notify(0x81) on the control method battery device. The Windows control method battery driver then evaluates the _BIX method and returns the updated information to the power manager.

The platform interrupts the ACPI firmware environment through the System Control Interrupt (SCI) in the case of an Embedded Controller-equipped platform and through a GPIO in the case of platforms with battery subsystem hardware connected directly to the core silicon.

ACPI operation with an embedded controller

Platforms that have their battery and power subsystem connected to the typical embedded controller use the ACPI Embedded Controller operation region to facilitate communications between the ACPI control method environment and the platform hardware.

The ACPI firmware must define the embedded controller in the ACPI namespace as described in section 12.11.1 of the ACPI specification, including:

- A Device() node for the embedded controller.
- A _HID object that indicates the device is an embedded controller.
- A _CRS object to denote the IO resources for the embedded controller.
- A _GPE object that defines the SCI for the embedded controller.
- An operation region describing the information contained within the embedded controller that can be accessed by other ACPI control method code in the namespace, including battery status and information methods.

Complete details are described in section 12 of the ACPI specification.

Accessing battery information from the embedded controller

The ACPI control method accesses the information from the embedded controller by reading the values described in the embedded controller's operation region.

Notifying the operating system when battery state changes

When the embedded controller detects a change in battery state, including a change in charging state or remaining capacity as specified by _BTP, the embedded controller

generates an SCI and sets the SCI_EVT bit in the embedded controller status command (EC_SC) register. The Windows ACPI driver will communicate with the embedded controller and issue a query command (QR_EC) to request specific information about the notification to be issued. The embedded controller then sets a byte value corresponding to the _QXX method to be executed. For example, the embedded controller and ACPI firmware can define value 0x33 to be an update to the battery status information. When the embedded controller sets the value 0x33 as the notification, the ACPI driver will execute the _QXX method. The contents of the _QXX method will typically be a Notify(0x80) on the control method battery device in the namespace.

ACPI operation with an I²C connected charging system

Platforms may also connect their battery and power subsystem connected to the core chipset through a low-power serial bus such as I²C. In these designs, the ACPI GenericSerialBus operation region is used to communicate between the ACPI control methods and battery subsystem hardware. Connecting the battery subsystem hardware to a GPIO interrupt allows the ACPI control methods to be executed when a battery status changes.

When the battery and power subsystem hardware is connected through I²C, the ACPI firmware must define:

- A Device() node for the GPIO controller device to which the I²C interrupt is connected, including:
  - _HID object describing the hardware ID of the GPIO controller.
  - _CSR object describing the interrupt and hardware resources of the GPIO controller.
  - _AEI object which maps one or more GPIO lines to ACPI event method execution. This allows ACPI methods to be executed in response to GPIO line interrupts.

- A Device() node for the I²C controller to which the battery fuel gauge and charging hardware are connected, including:
  - _HID and _CSR objects describing the hardware ID and resources of the I²C controller.
  - A GenericSerialBus OperationRegion within the scope of the I²C device describing the virtual command registers for the I²C device.
  - Field definitions within the GenericSerialBus OperationRegion. The field definitions allow ASL code outside of the I²C device to access the virtual command registers for the I²C device.

Describing the GPIO controller and mapping of GPIO lines to ACPI events allows the control methods for battery status and notification to be executed when a GPIO interrupt from an I^2C device is raised. Describing the GenericSerialBus operation region allows the ACPI code for battery status to communicate over the I^2C bus and read registers and information from the battery fuel gauge and charging subsystem.

Accessing battery information from the charging system

The battery status can be executed by ACPI control methods by sending and receiving commands over the I^2C bus to which the battery subsystem hardware is connected. The control method code backing the status and battery static information methods reads and writes data from the GenericSerialBus operation regions described in the ACPI namespace. The control method code reads the data from the fuel gauge device or static information about the battery capacity and cycle counts over the I^2C bus through the GenericSerialBus operation region.

Notifying Windows when the battery state changes

An interrupt can be generated by the battery subsystem hardware when the state changes and the interrupt is physically connected to a GPIO line on the core silicon. The GPIO line can be mapped to a specific control method execution using the _AEI object under the GPIO controller described in ACPI. When the GPIO interrupt occurs, the Windows ACPI subsystem runs the method associated with the specific GPIO line which can in turn do a Notify() on the control method battery device, causing Windows to re-evaluate the status and static information methods to update the battery status.

ACPI implementation of power supply object

The ACPI firmware must implement an ACPI power source device. This object must report itself with a hardware ID (_HID) of "ACPI0003". This object must also implement the ACPI _PSR (Power Source) method. This method returns the status of the power source and conveys if the power source is currently online (AC power) or offline (on battery power). All input power sources for the system must be multiplexed through a single _PSR method. E.g., _PSR must convey online if the system is powered through a DC barrel connector or a separate dock connector. Do not use multiple ACPI Power Source Devices.

The _PSR method must only report online (AC power) when the system is connected to mains power. When the state of _PSR changes, the platform must generate an interrupt and a Notify(0x80) on the device in the ACPI namespace. This must be performed immediately after the physical state change is detected by the platform.

ACPI implementation of battery static information

The ACPI firmware must implement the ACPI _BIX method for each battery that provides static information about the battery, including design capacity, cycle count, and serial number. The table below expands on the definitions of the fields described in the ACPI specification and enumerates Windows-specific requirements for this information.

Field	Description	Windows-specific requirements
Revision	Indicates _BIX revision	Must be set to 0x0
Power Unit	Determines the units reported by the hardware. Either: MA/MAh or mW/mWh.	Must be set to 0x0 to indicate units is mW/mWh
Design Capacity	Indicates the original capacity of the battery in mWh	Must be set to an accurate value and cannot be 0x0 or 0xFFFFFFFF
Last Full Charge Capacity	Indicates the current full charge capacity of the battery	Must be set to an accurate value and cannot be 0x0 or 0xFFFFFFFF This value must update each time cycle count increases.
Battery Technology	Indicates if the battery is rechargeable or one-time use.	Must be set to 0x1 to indicate the battery is rechargeable
Design Voltage	Indicates the design voltage of the battery	Must be set to the design voltage of the battery when new in mV. Must not be set to 0x0 or 0xFFFFFFFF.
Design capacity of Warning	Indicates an OEM-provided low battery warning level.	This value is ignored by Windows.
Design capacity of Low	Indicates the critical battery level at which Windows must immediately Shutdown or Hibernate before the system powers off.	Must be set to a value between 0x0 and 5% of the battery design capacity.
Battery Capacity Granularity 1	Indicates the minimum amount of remaining charge change that can be detected by the hardware between the Design Capacity of Warning and Design Capacity of Low.	Must be set to a value no larger than 1% of the battery design capacity.

Field	Description	Windows-specific requirements
Battery Capacity Granularity 2	Indicates the minimum amount of remaining charge change that can be detected by the hardware between Last Full Charge Capacity and Design Capacity of Warning.	Must be set to a value no larger than 75mW (approximately .25% of a 25Whr battery) which is (1/400) of the battery design capacity.
Cycle Count	Indicates the battery cycle count.	Must be set to a value larger than 0x0. Must not be set to 0xFFFFFFFF.
Measurement Accuracy	Indicates the accuracy of the battery capacity measurement.	Must be set to 95,000 or better, indicating 95% accuracy or better.
Max Sampling Time	The maximum supported sampling time between two successive _BST evaluations which will show a difference in remaining capacity.	No specific requirement.
Min Sampling Time	The minimum supported sampling time between two successive _BST evaluations which will show a difference in remaining capacity	No specific requirement.
Max Averaging Interval	The maximum averaging interval, in milliseconds, supported by the battery fuel gauge.	No specific requirement.
Min Averaging Interval	The minimum averaging interval, in milliseconds, supported by the battery fuel gauge.	No specific requirement.
Model Number	OEM-provided battery model number	Must not be NULL.
Serial Number	OEM-provided battery serial number	Must not be NULL.
Battery Type	OEM-provided battery type information	No specific requirement.
OEM Information	OEM-provided information	No specific requirement.

ACPI implementation of battery real-time status information

The ACPI firmware must implement the ACPI _BST method for each battery which provides real-time status information about the battery, including remaining capacity

and current rate of drain. The table below expands on the definitions of the fields described in the ACPI specification and enumerates the Windows-specific requirements for this information.

Field	Description	Windows-specific requirements
Battery State	Indicates if the battery is currently being charged, is discharging or is in a critical state.	The Battery State must report charging only if the battery is charging. Likewise, the Battery State MUST report discharging only if the battery is discharging. A battery that is neither charging nor discharging must report neither bit.
Battery Present Rate	Provides the current rate of drain in mW from the battery.	Must be greater than 0x0 and less than 0xFFFFFFFF. Must be accurate within the value of Measurement Accuracy in _BIX.
Battery Remaining Capacity	Provides the remaining battery capacity in mWh.	Must be greater than 0x0 and less than 0xFFFFFFFF. Must be accurate within the value of Measurement Accuracy in _BIX
Battery Present Voltage	Indicates the current voltage across the terminals of the battery.	Must be between a value of 0x0 and 0xFFFFFFFF in mV.

When any data in _BST changes, the platform must generate an interrupt and a Notify(0x80) on the battery device in the ACPI namespace. This must be performed immediately after the physical state change is detected by the platform. This includes any change in the Battery State field for the charging (i.e. Bit0) or discharging (i.e. Bit1) bits.

Additionally, the platform must implement the _BTP-Battery Trip Point-method. _BTP allows Windows to specify a remaining capacity threshold that when crossed, the platform must generate an interrupt and a Notify(0x80) on the battery device in the ACPI namespace. The _BTP method prevents Windows from needing to poll the battery periodically.

Battery control methods

The ACPI specification affords for device and operating system-specific control methods through the Device-Specific Method or _DSM control method. _DSM is described in section 9.14.1 of the ACPI specification.

Windows supports the following _DSM methods for control-method battery devices.

Thermal charge rate direction

Field	Value	Description
UUID	4c2067e3-887d-475c-9720-4af1d3ed602e	GUID indicating extensions to the Windows control method battery driver support
Revision ID	0x0	First revision of this capability
Function Index	0x1	Set battery charge throttle
Arguments	Thermal Limit	Integer value from 0 to 100 indicating the thermal charge limit. A value of 40% indicates the battery should be charged at 40% of maximum rate. A value of 0% indicates battery charging should be stopped until this method is called again.
Return Value(s)	None	n/a

User serviceable battery

Field	Value	Description
UUID	4c2067e3-887d-475c-9720-4af1d3ed602e	GUID indicating extensions to the Windows control method battery driver support
Revision ID	0x0	First revision of this capability
Function Index	0x2	Indicates this _DSM is for OSPM to determine if the battery device is user-serviceable or not.
Arguments	None	No arguments required.
Return Value(s)	Package containing a single integer.	0x0 if the battery is not user-serviceable and cannot be replaced by the end user, or can be replaced by the end user with additional tools. 0x1 if the battery can be replaced by the end-user without additional tools.

Charging watchdog required

Field	Value	Description
UUID	4c2067e3-887d-475c-9720-4af1d3ed602e	GUID indicating extensions to the Windows control method battery driver support
Revision ID	0x0	First revision of this capability
Function Index	0x3	Indicates this _DSM is for the OSPM to determine if the control method battery requires periodic watchdog resetting to maintain high current charging and the period at which the watchdog must be reset
Arguments	None	No arguments required.
Return Value(s)	Package containing a single integer.	0x0 if the battery does not require watchdog servicing. Values inclusive of 0x0000001e and 0x12C indicate the maximum poling interval in seconds. All other values are ignored and are treated as 0x0 and watchdog resetting is not required. If a valid watchdog interval is specified, Windows will execute the _BST method at an interval no longer than the watchdog value specified whenever the value of BatteryState in the _BST method is set to charging. Dynamic update of this value is not supported.

3rd Party Battery Miniport Drivers

In Windows 10, OEMs and IHVs can develop their own 3rd party battery miniport drivers to replace the Microsoft cmbatt.sys driver and communicate directly with the battery hardware. A sample battery driver is provided by Microsoft on GitHub and as part of the WDK samples kit .

USB charging (Windows 10 for desktop editions)

Microsoft recognizes the value in providing the option to support USB charging of a mobile device. With standardization efforts such as the EU's move to standardize mobile phone chargers, USB chargers are widely available and work across a wide variety of devices including Windows Phones, MP3 players, GNSS devices, etc. Microsoft understands the value of offering a single charger that can be used to charge multiple devices including a device running Windows. Further, given the broad industry support

for USB charging there are auxiliary benefits that reduce costs and environmental impact.

Beginning with Windows 8, a mobile device could be powered and/or charged through USB provided that the battery charging requirements outlined below are met. In addition, there are a number of USB-specific requirements that must be met to ensure a quality user experience.

1. USB power/charge must be implemented entirely in the platform firmware. Support must not require an operating system, driver, or application.

2. The device MUST NOT enumerate when connected to another device. As a result, the device will not charge when connected to a standard PC USB port as these ports are limited to 500mA by default. The only exceptions are when this port is used for debugging and for initial factory firmware programming.

3. The device supports charging from a dedicated USB charging port. The device must charge when connected to a charger that is compliant with the USB Battery Charging specification version 1.2. The device should not draw more than 1.5A per charging standards when connected to a standard USB charger. The OEM may opt to support higher current levels provided the following conditions are met:

    - The device automatically detects the charger type and charges at the appropriate rate for the specific charger type.
    - The device and charger meet all relevant electrical and safety standards.
    - The OEM ships the charger and associated cable with the device.

4. USB charging is supported over either a standard micro-AB receptacle, USB-C (recommended) or a proprietary dock connector. A micro-B receptacle is NOT allowed on the device. If using a proprietary dock connector, the OEM must ship an appropriate cable with the device to enable charging from a standard USB charger.

5. If the micro-AB port is implemented, the device must automatically detect the cable type, configuration, and assume the appropriate role. If a micro-B plug is inserted and debugging is not enabled on the port, the charger role should be assumed. If a micro-B plug is inserted and debugging is enabled on the port, the debug role should be assumed (i.e. charging is not supported). If a micro-A plug is inserted, the USB host role is assumed where attached USB devices is recognized by Windows.

6. If the micro-AB port also functions as a debug port, the device must provide a means through firmware to switch between the charger and debug role. The

default setting as shipped to the end user must have debug DISABLED.

7. If the micro-AB port also functions as a debug port, the device should provide an alternative input power path through either a dedicated barrel connector or proprietary dock connector.

Platform designer and implementer checklists

You can use the following checklists to validate the platform design and system firmware adhere to the battery and charging subsystem guidance outlined.

Battery subsystem and ACPI firmware implementation checklist

System designers should ensure they have completed the following tasks in their ACPI firmware to ensure correct reporting of battery and power subsystem information to Windows:

- Add a Device() object for each battery device in the ACPI namespace.
- Each battery device must provide the following control methods and objects:
  - _HID with a value of PNP0C0A.
  - _BIX-Battery Information Extended:

    Conveys the battery static information including the last full charge capacity, design capacity and cycle count.

  - _BST-Battery Status:

    Conveys the current battery status, including the remaining capacity, rate of drain, and charging state.

  - _BTP-Battery Trip Point:

    Enables an event-driven battery status model to reduce periodic work for polling. _BTP allows Windows to specify a threshold of remaining charge capacity at which the platform should issue at Notify(0x80) on the battery device to request Windows to update the battery status information.

  - _STA-General Status:

    Allows Windows to know if the battery is present in the device where a battery may be removable or where there may be a battery in a portable dock.

- Add a single Device() object for an AC adapter / Power Source in the ACPI namespace.
- The power source device must provide the following control methods and objects:
  - _HID with a value of ACPI0003
  - _PSR-Power Source:

    Conveys if the power source is currently online (AC power) or offline (on battery power). All input power sources for the device must be multiplexed through the _PSR method. For example, the _PSR must convey online if the device is powered through a DC barrel connector or separate dock connector. Do not use multiple ACPI Power Source Devices.
- The _BIX method must support the fields and constraints described in the battery static information above:
  - The *Revision* field must be set to 0x0.
  - The *Power Unit* field must be set of 0x0.
  - The *Design Capacity* and *Last Full Charge Capacity* values must be set to accurate values from the battery and charging subsystem and not set equal to 0xFFFFFFFF or 0x00000000.
  - The *Battery Technology* field must be set to 0x1.
  - The *Design Voltage* field must be set accurately and not equal to 0x00000000 or 0xFFFFFFFF.
  - The *Design Capacity of Low* must be set to the minimum value required to Hibernate or Shutdown the system from a fully on state.
  - The *Battery Capacity Granularity 1* and *Battery Capacity Granularity 2* fields must be set to a value no larger than 1% of the battery design capacity.
  - The *Cycle Count* field must be accurately filled in from the battery subsystem.
  - The *Measurement Accuracy* field must be set to 80,000d or better.
  - The *Model Number and Serial Number* fields must not be set to NULL.
- Provide a _BST method that allows Windows to poll for real-time battery status. The fields in the _BST method must all be returned dynamically from the underlying power and battery charging subsystem. Their accuracy must be within the value of Measurement Accuracy in the _BIX method.
- Provide a _BTP method that allows Windows to specify a remaining charge capacity threshold at which the platform will interrupt Windows with a Notify(0x80) on the battery device.

- Ensure a Notify(0x80) is only issued in response to a battery status change or _BTP charge capacity limit trip. Do not execute a Notify(0x80) periodically.
- When the battery level reaches the value specified in _BIX.DesignCapacityofLow, the platform must generate a Notify(0x80) on the control method battery device.
- For systems with multiple batteries, fully implement a control method battery device for each battery.
  - The first battery in the namespace should be the primary battery for the system to aid in debugging purposes.
- Implement the _DSM method under each battery device to indicate if the battery is user-serviceable.
- Implement the _DSM method if a periodic watchdog reset is required during charging and Windows will guarantee periodic execution of the _BST method within that polling window.
- Implement the _DSM method if battery charging rate control is required for the thermal model on the platform.

Battery Saver

Article • 06/24/2024

The battery saver feature exists on Windows 11 24H1 (and earlier) and on Windows 10. It is replaced by Energy Saver on Windows 11 24H2 and later. It helps conserve power when a system is running on battery. When battery saver is on, some Windows features are disabled or behave differently.

Users can choose to enable battery saver when the battery level reaches a certain percentage. By default, battery saver will turn on automatically when your battery falls below 20%.

When battery saver is on, several Windows features behave differently:

- Microsoft Store Mail, People, and Calendar apps do not sync.
- Apps running in the background are blocked. Users can allow specific individual apps to run while in battery saver mode. Certain categories of apps continue to run. For example, VOIP apps are not blocked.
- Non-critical Windows update downloads are blocked. However, Windows update scans still occur.
- Display brightness is reduced by 30%. Users can enable or disable this setting.
- OEMs can enable configure the display brightness value on Windows. This value is controlled in the ESBRIGHTNESS SUB_ENERGYSAVER Powercfg setting.
- The majority of telemetry is blocked.
    - Only critical telemetry is uploaded
    - Census data, and similar telemetry is critical and will be uploaded
- Windows task scheduler tasks trigger only if the task is:
    - Not set to **Start the task only if the computer is idle...** (task doesn't use **IdleSettings**)
    - Not set to run during automatic maintenance (task doesn't use **MaintenanceSettings**)
    - Is set to **Run only when user is logged on** (task **LogonType** is **TASK_LOGON_INTERACTIVE_TOKEN** or **TASK_LOGON_GROUP**)

All other triggers are delayed until the system exits battery saver mode.

For more information about accessing battery saver status in your application, see SYSTEM_POWER_STATUS.

Energy Saver

Article • 06/24/2024

Energy saver replaces battery saver and exists on Windows 11 24H2 and later versions. It significantly extends battery life and reduces energy consumption by automatically managing power usage across your system and apps for an optimal balance of PC performance and energy efficiency.

Users can choose to enable energy saver in several different ways.

- Users can conveniently toggle on energy saver via Quick Settings in the system tray at any time. To maximize the device's battery life, Windows recommends users to enable energy saver all the time, even if the device battery still has plenty of charge left.
- Users can configure energy saver in Windows Settings to run automatically whenever their devices reach a certain battery percentage.
- Users can configure their devices in Windows Settings to always run energy saver when their devices are plugged in.

When energy saver is on, several Windows features behave differently:

- Users cannot change the power mode in Windows Settings.
- Display brightness is reduced by 30%. Users can enable or disable this setting.
  - OEMs can enable and configure the display brightness value. This value is controlled in the ESBRIGHTNESS SUB_ENERGYSAVER Powercfg setting.
- Transparency effects on window backgrounds is disabled.

When a device is unplugged and on low battery, these additional behaviors can be expected when energy saver is enabled:

- Certain Microsoft apps, including OneNote, OneDrive, Phone Link, etc. may not sync.
- Apps running in the background are blocked. Users can allow specific individual apps to run while in energy saver mode. Certain categories of apps continue to run. For example, VOIP apps are not blocked.
- Non-critical Windows update downloads are blocked. However, Windows update scans still occur.

- The majority of telemetry is blocked.
  - Only critical telemetry is uploaded
  - Census data, and similar telemetry is critical and will be uploaded
- Windows task scheduler tasks trigger only if the task is:
  - Not set to **Start the task only if the computer is idle...** (task doesn't use **IdleSettings**)
  - Not set to run during automatic maintenance (task doesn't use **MaintenanceSettings**)
  - Is set to **Run only when user is logged on** (task **LogonType** is **TASK_LOGON_INTERACTIVE_TOKEN** or **TASK_LOGON_GROUP**)

  All other triggers are delayed until the system exits energy saver mode.

The list is subject to further changes as the devices get updates from Windows Update.

For more information about accessing energy saver status in your application, see SYSTEM_POWER_STATUS.

Windows power and battery subsystem requirements

Article • 05/18/2022

To meet user expectations and provide a quality experience, all Windows PCs must have consistent behavior for battery charging and system power transitions. Starting with Windows 8 and Windows 8.1, this design principle extends to battery-powered hardware platforms that are based on System on a Chip (SoC) integrated circuits and that support the modern standby power model.

This section describes the user experience requirements for the power and battery charging subsystem in a hardware platform that runs Windows. It provides detailed implementation guidance for platform designers and firmware developers. They can use this information to ensure their platforms are compatible with Windows power management software.

The information in this section pertains to all processor architectures—x86, x64, and Arm—that are supported by Windows. In addition, this information applies both to low-power platforms that implement the modern standby power model and to platforms that support the traditional ACPI sleep (S3) and hibernate (S4) power states.

In this section

Topic	Description
User experience principles for battery charging	Outlines the five fundamental user experience principles for battery charging.
Power and charging subsystem implementation	Describes how to implement the battery power and charging subsystem on a Windows platform.
ACPI battery and power subsystem firmware implementation	This topic details how the platform should expose power subsystem information to the Windows power manager.
Battery and power subsystem hardware design	Explains the many factors in designing the battery and power subsystem hardware.
Platform requirements and motivation	Describes the hardware requirements and motivation for the power and battery subsystem on the Windows platform.

Topic	Description
Power and battery subsystem checklist	System designers can use the checklists included in this section to verify that their platform designs and system firmware follow the Windows guidelines for power and battery subsystem operation.

User experience principles for battery charging

Article • 02/11/2020

Users expect to have a consistent and quality experience around battery charging. Therefore, all systems in the Windows ecosystem must have a consistent battery charging experience, regardless of form factor, instruction set, or platform architecture.

From a user experience perspective, there are five fundamental principles for battery charging:

- **Charging always occurs when connected to the charger.** The system always charges the battery when it is connected to the charger, except in cases of battery failure or thermal conditions. The term charger is used to describe the AC adapter or power brick that is included with the system.
- **Windows can always boot when connected to the charger.** If the system is shut down (ACPI S5 state), it is always able to boot into Windows when connected to the charger provided with the system. Furthermore, the system must boot even if the battery is not present, as can be the case for a system with a removable battery.
- **Charging is managed autonomously by the hardware.** Battery charging in a Windows PC is performed independently by the system hardware and does not require intervention by firmware, drivers, Windows, or other software running on the main CPU. Battery charging continues even when the user completely shuts down the system, and during the rare instances in which Windows encounters a fatal error.
- **Charging stops automatically when the battery is fully charged or when a fault occurs.** The system hardware automatically stops charging when the battery is completely charged. (The term completely charged refers to the level of charge at which the OEM considers the battery to be fully charged. For reasons of battery maintenance and lifecycle, this state may not be 100 percent of the cell's capacity.) This is done without requiring firmware, Windows, drivers, or other software running on the main CPU(s). Charging is also automatically halted in the case of a battery or thermal fault condition.
- **Windows does not automatically boot when power is applied.** This requirement does not apply when the system is in a test lab environment, as described in Automated lab testing considerations. When the system is in the off state (ACPI S5 state), it does not automatically boot when power is applied to the system. The system remains in the off state until it is manually powered on by the user. This applies to UEFI and Windows boot.

Note Systems that can only charge through a Universal Serial Bus (USB) connector might not be able to support all of these user-experience goals. These systems might require booting into firmware or Windows to interrogate the USB port and determine the correct charge rate.

Power and charging subsystem implementation

Article • 12/05/2019

The battery power and charging subsystem in a Windows platform should implement the ACPI Control Method Battery interface. Windows uses this interface to get status information from the battery subsystem and to receive notifications of battery events.

Windows battery and power subsystem driver model

Windows features a robust battery and power subsystem device driver model. Power management information is conveyed to the Windows power manager through a battery device driver. This information is then aggregated and exposed to the Windows user interface through battery device IRPs and a set of power management software APIs.

The battery driver model for Windows is a port/miniport model; that is, the battery model and interfaces are defined so that new battery types can be exposed through a miniport driver, as shown in the following block diagram. However, in practice, only the ACPI Control Method Battery interface is used. A Human Interface Device (HID) battery miniport driver for USB-attached Uninterruptible Power Supply (UPS) devices is supported, but this driver is not used for system batteries.

All Windows platforms are expected to expose their batteries and charging subsystem through the ACPI Control Method Battery interface. The battery miniport interface should not be used for platform-specific battery charging subsystems. The ACPI 5.0 specification defines control methods to enable Windows to obtain battery information and current status. In addition, the ACPI interface provides events to enable the hardware platform to notify Windows of battery and power source changes, such as a transition from AC to battery power.

Obtaining battery status

The Windows power manager periodically requests status information from the battery, including the remaining charge capacity and the current rate of drain. This request originates in the power manager itself, a higher-level user interface component, or in an application. The power manager turns the request into an I/O request packet (IRP) to

the battery device, which is then turned into the appropriate control method by the Windows control-method battery driver, Cmbatt.sys. The method executes and the result is returned up the driver stack.

In the case of status information, the battery status (_BST) method is the appropriate control method. The _BST method requires the ACPI firmware to obtain current information from the power subsystem. The result of the _BST method is a buffer that contains the required information, which is returned to the control-method battery driver. The control-method battery driver converts the buffer contents to the format required by the battery driver and Windows power manager.

State change notifications

The power and battery subsystem generate several notifications to the operating system for state changes, including changes in the level of charge, transitions from AC to battery power, and so on. It is undesirable to continuously poll for these state changes because doing so requires a high polling frequency. Therefore, the hardware platform must implement an event-driven model to notify Windows of changes in the power and battery subsystem.

When battery status (including remaining capacity or charging status) changes, the ACPI firmware must issue a Notify(0x80) command on the Control Method Battery device. In response, the Control Method Battery driver in Windows evaluates the _BST method and returns the updated information to the power manager.

When battery static data changes (including last full charge capacity, design capacity, and cycle count), the ACPI firmware is required to issue a Notify(0x81) command on the Control Method Battery device. In response, the Control Method Battery driver in Windows evaluates the _BIX method and returns the updated information to the power manager.

ACPI Notify events are signaled through either the System Control Interrupt (SCI), in the case of embedded controller-equipped platforms, or through general-purpose I/O (GPIO), in the case of platforms that have battery subsystem hardware directly connected to the core silicon.

ACPI battery and power subsystem firmware implementation

Article • 01/11/2022

Windows platforms are expected to expose their battery devices and AC adapters in firmware through ACPI by using standardized control method interfaces, as described in the ACPI specification.

Every mobile Windows platform must have one power source device and one or more batteries. Information from these subsystems is used to convey power status to the user. This status includes such information as whether the platform is running on AC or DC power, the level of charge for each battery, and battery charging status. The Windows power manager aggregates this information and makes it available to the Windows battery meter and other power management components.

This topic details how the platform should expose power subsystem information to the Windows power manager. For more information, see Chapter 10, Power Source Devices, in the ACPI specification.

> **Note**
>
> Some of the information described in this article is specific to Windows and is not detailed in the recent ACPI specification.

ACPI power supply object

The ACPI firmware must provide and implement a single ACPI power source device per section 10.3 of the ACPI specification. This object must report itself with a Hardware ID (_HID) of "ACPI0003".

Note For systems that have multiple power sources, all physical power sources must be multiplexed through a single power device object in ACPI. This object must represent the composite status of power inputs for the system. Client systems must not provide multiple power source device objects. (Additional power source objects may be present on server systems that have multiple power supplies.)

This object must also implement the Power Source (_PSR) method. This method returns the status of the power source and conveys whether the power source is currently online (AC power) or offline (on battery power). The _PSR method must only report

online (AC power) when the system is connected to main power. When the state of _PSR changes, the platform must generate an interrupt and a Notify(0x80) command on the device in the ACPI namespace. This must be performed immediately after the platform detects the physical state change.

ACPI battery control methods

The ACPI firmware must provide and implement an ACPI control method object for each battery in the system per section 10.2 of the ACPI specification. Each battery device must do all of the following:

- Identify itself with a Hardware ID (_HID) of "PNP0C0A".
- Implement the Status (_STA) method to indicate if the device is enabled, disabled, or not present.
- Report static information by using the Battery Information Extended (_BIX) method.
- Report battery status by using the Battery Status (_BST) control method.
- Support level-of-charge events by using the Battery Trip Point (_BTP) mechanism.

Optionally, a battery device can implement the Slot Unit Number (_SUN) or indicate battery order for the purposes of display in the user interface (UI).

The following discussion presents the details of these methods and describes their Windows-specific requirements.

ACPI implementation of battery static information

The ACPI firmware must implement the _BIX method for each battery to provide static information about the battery, including design capacity, cycle count, and serial number. The following table expands on the definitions of the fields that are described in section 10.2.2.2 of the ACPI specification, and enumerates Windows-specific requirements for this information.

Topics	Description	Windows-specific requirements
Revision	Indicates _BIX revision.	Must be set to 0x0.
Power unit	Determines if the units reported by the hardware are milliamps and milliamp-hours or milliwatts and milliwatt-hours.	Must be set to 0x0 to indicate units are milliwatts and milliwatt-hours. This value must not change at runtime.

Topics	Description	Windows-specific requirements
Design capacity	Indicates the original capacity of the battery in milliwatt-hours.	Must be set to an accurate value and cannot be set to 0x0 or 0xFFFFFFFF. This value must not change at runtime.
Last full battery charge	Indicates the current full charge capacity of the battery.	Must be set to an accurate value and cannot be set to 0x0 or 0xFFFFFFFF. This value must update each time cycle count increases. This value must remain constant when the battery is discharging. We recommend that this value be updated only when the battery reaches full charge.
Battery technology	Indicates if the battery is rechargeable or one-use.	Must be set to 0x1 to indicate the battery is rechargeable.
Design voltage	Indicates the design voltage of the battery.	Must be set to the design voltage of the battery when new in millivolts. Must not be set to 0x0 or 0xFFFFFFFF. This value must not change at runtime.
Design capacity of Warning	Indicates an OEM-provided low battery warning level.	Windows ignores this value.
Design capacity of Low	Indicates the critical battery level at which Windows must immediately shut down or hibernate before the system powers off.	Must be set to a value between 0 and 5 percent of the battery design capacity.
Battery capacity granularity 1	Indicates the minimum amount of remaining charge change that can be detected by the hardware between the Design Capacity of Warning and Design Capacity of Low.	Must be set to a value no larger than 1 percent of the battery design capacity.
Battery capacity granularity 2	Indicates the minimum amount of remaining charge change that can be detected by the hardware between Last Full Charge Capacity and Design Capacity of Warning.	Must be set to a value no larger than 75 milliwatts (approximately 0.25 percent of a 25 watt-hour battery). (1/400) of the battery design capacity.
Cycle count	Indicates the battery cycle count. Must be set to a value larger than 0x0.	Must not be set to 0xFFFFFFFF.

Topics	Description	Windows-specific requirements
Measurement accuracy	Indicates the accuracy of the battery capacity measurement.	Must be set to 95,000 or better, indicating 95 percent accuracy or better.
Max sampling time	The maximum supported sampling time between two successive _BST evaluations, which will show a difference in remaining capacity.	No specific requirement.
Min sampling time	The minimum supported sampling time between two successive _BST evaluations, which will show a difference in remaining capacity.	No specific requirement.
Max averaging interval	The maximum averaging interval, in milliseconds, supported by the battery fuel gauge.	No specific requirement.
Min averaging interval	The minimum averaging interval, in milliseconds, supported by the battery fuel gauge.	No specific requirement.
Model number	OEM-provided battery model number.	Must not be NULL.
Serial number	OEM-provided battery serial number.	Must not be NULL.
Battery type	OEM-provided battery type information.	No specific requirement.
OEM information	OEM-provided information.	No specific requirement.

In addition to these requirements, the platform firmware must generate an interrupt and a Notify(0x81) command on the battery device in the ACPI namespace whenever any of the battery state data in _BIX changes. This includes last full charge capacity, design capacity, and cycle count. This must be performed immediately after the state change is detected by the platform.

The last full charge capacity represents the estimated amount of energy the battery is expected to hold the last time the battery was fully charged. Windows assumes that this value is updated only when the battery has been charged. Therefore, the last full charge

capacity value must not change while the battery discharges. We recommend that this only be updated when the battery reaches full charge.

ACPI implementation of battery real-time status information

The ACPI firmware must implement the _BST method for each battery to provide real-time status information about the battery, including remaining capacity and current rate of drain. The following table expands on the definitions of the fields that are described in section 10.2.2.6 of the ACPI specification, and enumerates Windows-specific requirements for this information.

Topics	Description	Windows-specific requirements
Battery state	Indicates if the battery is currently being charged, is discharging, or is in a critical state.	The Battery State must report charging only if the battery is charging. Likewise, the Battery State must report discharging only if the battery is discharging. A battery that is neither charging nor discharging must report neither bit.
Battery present rate	Provides the current rate of drain in milliwatts from the battery.	Must be a value greater than 0x0 and less than 0xFFFFFFFF. Must be accurate within the value of Measurement Accuracy in _BIX.
Battery remaining capacity	Provides the remaining battery capacity in milliwatt-hours.	Must be greater than 0x0 and less than 0xFFFFFFFF. Must be accurate within the value of Measurement Accuracy in _BIX.
Battery present voltage	Indicates the current voltage across the terminals of the battery.	Must be between a value of 0x0 and 0xFFFFFFFF in millivolts.

When any data in _BST changes, the platform must generate an interrupt and a Notify(0x80) on the battery device in the ACPI namespace. This must be performed immediately after the physical state change is detected by the platform. This includes any change in the Battery State field for the charging bit (Bit0) or discharging bit (Bit1).

Additionally, the platform must implement the _BTP-Battery Trip Point-method. _BTP allows Windows to specify a remaining capacity threshold that when crossed, causes the platform to generate an interrupt and a Notify(0x80) on the battery device in the ACPI namespace. The _BTP method prevents Windows from needing to poll the battery periodically.

Windows OS-specific battery control methods

The ACPI specification affords for device and operating system-specific control methods through the Device-Specific Method or _DSM control method. _DSM is described in section 9.14.1 of the ACPI specification.

Windows 8 supports the following _DSM methods for Control Method Battery devices.

Thermal charge rate direction

Topics	Value	Description
UUID	4c2067e3-887d-475c-9720-4af1d3ed602e	GUID that indicates extensions to Windows Control Method Battery driver support.
Revision ID	0	First revision of this capability.
Function index	0x1	Set battery charge throttle.
Arguments	Thermal limit	Integer value from 0 to 100 indicating the thermal charge limit. A value of 40 percent indicates the battery should be charged at 40 percent of maximum rate. A value of 0 percent indicates battery charging should be stopped until this method is called again.
Return value(s)	None	N/A

User-serviceable battery

Topics	Value	Description
UUID	4c2067e3-887d-475c-9720-4af1d3ed602e	GUID that indicates extensions to Windows Control Method Battery driver support.
Revision ID	0	First revision of this capability.
Function index	0x2	Indicates this _DSM is for OSPM to determine if the battery device is user-serviceable or not.
Arguments	None	No arguments are required.

Topics	Value	Description
Return value(s)	Package containing a single integer.	0x0 if the battery is not user-serviceable and cannot be replaced by the end user, or can be replaced by the end user with additional tools. 0x1 if the battery can be replaced by the end-user without additional tools.

Charging watchdog required

Topics	Value	Description
UUID	4c2067e3-887d-475c-9720-4af1d3ed602e	GUID that indicates extensions to Windows Control Method Battery driver support.
Revision ID	0	First revision of this capability.
Function index	0x3	Indicates this _DSM is for OSPM to determine if the Control Method Battery requires periodic watchdog resetting to maintain high current charging and the period at which the watchdog must be reset.
Arguments	None	No arguments are required.
Return value(s)	Package containing a single integer.	0x0 if the battery does not require watchdog servicing. Values inclusive of 0x0000001e and 0x12C indicate the maximum poling interval in seconds. All other values are ignored and are treated as 0x0 and watchdog resetting is not required. If a valid watchdog interval is specified, Windows will execute the _BST method at an interval no longer than the watchdog value specified when the value of BatteryState in the _BST method is set to charging. Dynamic update of this value is not supported.

Windows provides a detailed view of system batteries in the desktop tray application. Each battery, together with its current status, is listed in the UI. The following example UI shows two batteries.

The order in which Windows shows batteries can be specified by the firmware. This is achieved by using the Slot User Number (_SUN) method as defined in section 6.1.11 of the ACPI specification. The _SUN method returns an integer that represents the unique ID of the battery. Windows lists each battery object in ascending order based on the value of the _SUN method.

If any battery object includes a _SUN method, then all battery objects must also provide a _SUN method. Windows does not support the case where some batteries have a _SUN

method and other batteries do not. If no _SUN objects are provided, Windows sorts the batteries based on their full device instance path; this configuration is supported, but not recommended.

Battery and power subsystem hardware design

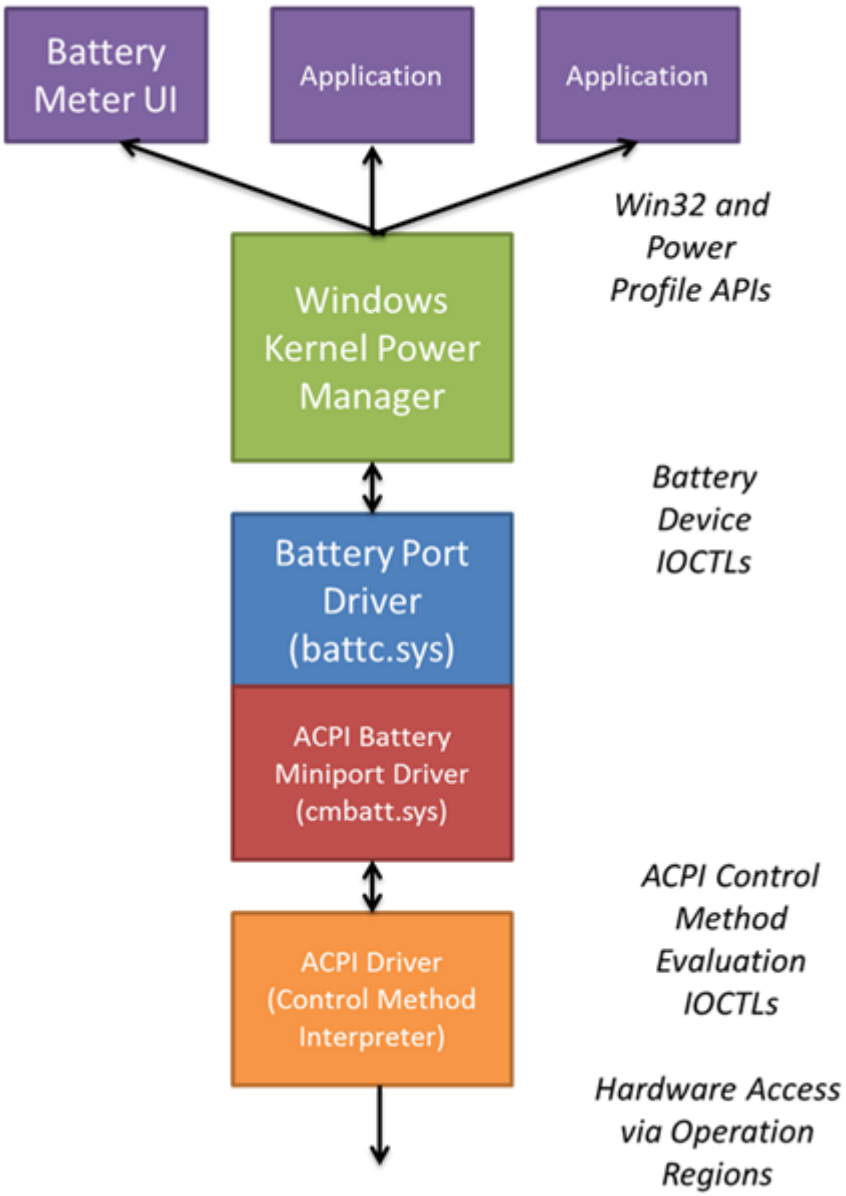
Article • 06/21/2022

Starting with Windows 8, system hardware designers can choose between two different hardware topologies for the battery and power subsystems in their Windows platforms.

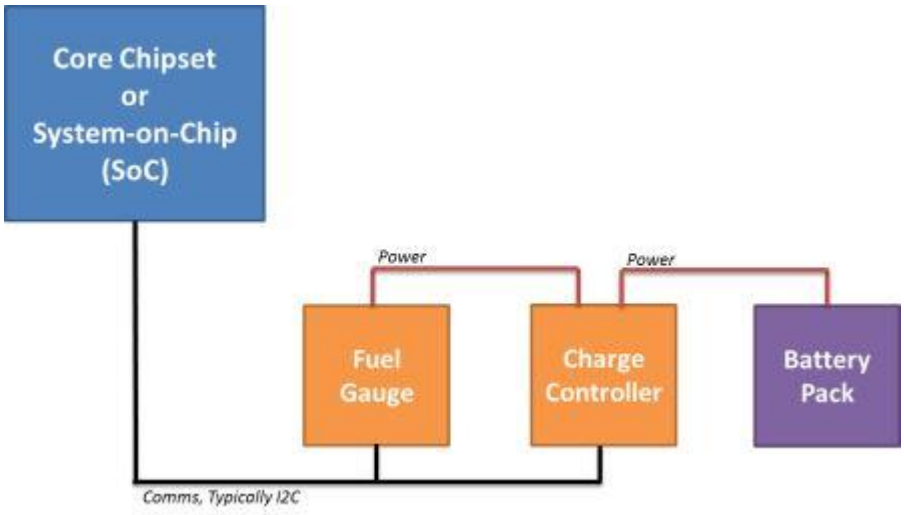
Hardware topologies

Generally, Windows expects one of two hardware topologies for the power and charging subsystem.

The first topology is shown in the following block diagram. This topology, which is common in PCs that run Windows 7, uses an embedded controller in the platform. The embedded controller typically performs multiple functions in a mobile Windows PC, including power source control, battery charge management, power button/switch detection, and PS/2-compatible keyboard and mouse input. The embedded controller is typically connected to the core silicon through the Low Pin Count (LPC) bus. Windows queries and is notified about power subsystem information through the ACPI embedded controller interface.



The second topology is shown in the following block diagram. This topology uses a battery charge controller and fuel gauge component, which are connected directly to the platform's core silicon over a simple peripheral bus, such as I^2C. In this configuration, Windows queries and is notified about power subsystem changes through communications over the I^2C bus. A Simple Peripheral Bus (SPB) operation region enables ACPI control method code in firmware to communicate to the battery charge controller and fuel gauge components that are connected to the core silicon over an I^2C bus.



ACPI operation with an embedded controller

Platforms that have their battery and power subsystem connected to the typical platform embedded controller use the ACPI embedded controller operation region to facilitate communications between the ACPI control method environment and the platform hardware.

The ACPI firmware must define the embedded controller in the ACPI namespace. This definition includes the following:

- A Device() node for the embedded controller.
- A _HID object that indicates the device is an embedded controller.
- A _CRS object to denote the I/O resources for the embedded controller.
- A _GPE object that defines the SCI for the embedded controller.
- An operation region that describes the information that is contained in the embedded controller that can be accessed by other ACPI control method code in the namespace, including battery status and information methods.

For more information, see section 12.11, "Defining an Embedded Controller Device in ACPI Namespace", in the ACPI 5.0 specification.

Accessing battery information from the embedded controller

ACPI control methods access information from the embedded controller by reading the values that are described in the embedded controller's operation region.

Notifying Windows when battery state changes (embedded controller)

When the embedded controller detects a change in battery state, including a change in charging state or remaining capacity as specified by _BTP, the embedded controller generates an SCI and sets the SCI_EVT bit in the embedded controller status command (EC_SC) register. The Windows ACPI driver, Acpi.sys, communicates with the embedded controller and issues a query command (QR_EC) to request specific information about the notification to be issued. The embedded controller sets a byte value corresponding to the _QXX method to be executed. For example, the embedded controller and ACPI firmware can define value 0x33 to be an update to battery status information. When the embedded controller sets the value 0x33 as the notification, Acpi.sys will execute the _QXX method. The _QXX method typically issues a Notify(0x80) command on the Control Method Battery device in the namespace.

Power consumption

Special care must be taken on modern standby systems to ensure that minimum battery life goals for modern standby are achieved. On modern standby systems, the nominal power consumed by the embedded controller for power and battery subsystem must be below 5 milliwatts. On PCs that use the traditional S3/S4 power states, make sure that the embedded controller does not impact battery-life goals. There are no specific nominal power requirements for systems that use S3/S4.

ACPI operation with an SPB-connected charging subsystem

Platforms can also connect the battery and power subsystem connected to the core chipset by using a low-power simple peripheral bus (SPB) such as I^2C. In these designs, the ACPI GenericSerialBus operation region is used to communicate between ACPI control methods and battery subsystem hardware. Connecting the battery subsystem hardware to a GPIO interrupt allows ACPI control methods to be executed when battery status changes.

When the battery and power subsystem hardware is connected by using I^2C, ACPI firmware must define the following:

- A Device() node for the GPIO controller device to which the I^2C interrupt is connected, including:
  - _HID object describing the hardware ID of the GPIO controller.

    - _CSR object describing the interrupt and hardware resources of the GPIO controller.
    - _AEI object that maps one or more GPIO lines to ACPI event method execution. This enables ACPI methods to be executed in response to GPIO line interrupts.
- A Device() node for the I^2C controller to which the battery fuel gauge and charging hardware are connected, including:
    - _HID and _CSR objects that describe the hardware ID and resources of the I^2C controller.
    - A GenericSerialBus operation region within the scope of the I^2C device that describe the virtual command registers for the I^2C device.
    - Field definitions in the GenericSerialBus operation region. The field definitions allow ASL code outside of the I^2C device to access the virtual command registers for the I^2C device.

Describing the GPIO controller and mapping of GPIO lines to ACPI events enables control methods for battery status and notification to be executed when a GPIO interrupt from an I^2C device is raised. Describing the GenericSerialBus operation region allows ACPI code for battery status to communicate over the I^2C bus and read registers and information from the battery fuel gauge and charging subsystem.

Accessing battery information from the charging subsystem

Battery status can be executed by ACPI control methods by sending and receiving commands over the I^2C bus to which the battery subsystem hardware is connected. The control method code backing the status and battery static information methods reads and writes data from GenericSerialBus operation regions described in the ACPI namespace. The control method code can read data from the fuel gauge device or static information about the battery capacity and cycle count over the I^2C bus through the GenericSerialBus operation region.

Notifying Windows when battery state changes (subsystem hardware)

The battery subsystem hardware can generate an interrupt when state changes, or from a GPIO line on the core silicon. The GPIO line can be mapped to a specific control method execution by using the _AEI object under the GPIO controller described in ACPI. When the GPIO interrupt occurs, the Windows ACPI subsystem executes the method associated with the specific GPIO line, which can in turn issue a Notify() command on

the Control Method Battery device. This causes Windows to re-evaluate status and static information methods to update battery status.

Power and charging indicators

Windows provides indication of power source and battery status in the operating system. This is presented to the user in several places, including the battery tray icon on the desktop, on the Start menu, and directly on the Lock screen.

Windows 8 platforms can also present a visible indicator to the user of charging status. The following figures show two UI examples. The indicator that is used must have little impact on power consumption and the user experience.

Windows power and charging user interface elements

Windows provides an indication of power source and charging status in three key locations:

- **On the lock screen.** A battery icon with power source and charge status is displayed.
- **On the time-and-date indicator when hovering over the Start button.** A battery icon with power source and charge status is displayed.
- **Battery icon on the desktop.** A battery icon with power source and charge status is displayed. More information is available when you click on the battery icon; this includes capacity remaining, estimated time remaining, and per-battery details if the system has multiple batteries

. For modern standby-capable platforms, if the system is in S0 and the lid (if present) is not closed, Windows briefly lights up the display when the system is connected to the charger and power is applied. This enables users to see the platform respond to the action of connecting the charger.

Platform hardware charging indicators

The user interface elements that are built into Windows address the scenarios where Windows is running and the display is visible to the user. However, these on-screen indicators are not visible when the system is shutdown, hibernated, sleeping, or otherwise not running.

A platform may include an LED to indicate that power is present. It is preferred that such an LED not be placed on the system chassis. Instead, the LED it should be either on the power brick, power cable, or power connector. Optionally, this LED can also indicate charge status to the user.

If an LED is provided, it should not vary in intensity or color over time or otherwise flash or blink, because that presents a distraction to users. It may however change color to indicate charge status; for example, yellow when charging, green when fully charged, or red when a failure occurs.

Real-time clock reserve battery capacity

Maintaining accurate time is essential to delivering a great user experience. In addition, accurate time is required to connect to services such as the Microsoft Store. All Windows systems should maintain accurate time for a period of at least four weeks, even when off. Typically this is done by employing a separate backup battery to maintain the real-time clock (RTC). This is not always possible or practical on highly portable form-factors.

System designers can use a dedicated battery or reserve a portion of the main system battery. Given the modest power requirements of the RTC, a relatively low reserve threshold will provide guarantees that match dedicated backup batteries that exist in today's PCs.

Design guidelines

OSPM provides a method for system designers to override the Windows OS critical battery event. When the battery gets to a critical level (in milliwatt-hours), as defined by the _BIX (Battery Information Extended – Design Capacity for Low) method in the Control Method Battery implementation, the firmware issues a Notify command to the OS. At that point, Windows will perform an emergency shutdown or hibernate to preserve system state.

All designs must meet the following requirements:

- **The Design Capacity for Low** in the _BIX method must be set to at least 675 milliwatt-hours of the full design capacity (in addition to the capacity necessary to reliably perform the critical action).
- The reserve capacity above must be less than four percent of the full design capacity.

Charging performance

The time required to fully charge the system battery is a concern to the user. Many systems are charged overnight or during other periods when the user is not interacting with the system. However, when the battery is fully drained and the user wants to use the system in a portable fashion, charging performance is a primary concern.

Windows recommends that all platforms be capable of charging the system battery from 5 percent to 90 percent within four hours or less when the system is booted and in modern standby with the screen off.

System designers should pay special attention to charge rate for systems that support Universal Serial Bus (USB)-based charging only. Systems that have USB charging only and large battery capacity might not meet the customer's expectation for charging performance.

If USB charging is required on platforms with large battery capacity (greater than 30 watt-hours), the system designer should also provide a high-power DC input and bundle a high-power DC charger with the system. This also enables the platform battery to be charged during interactive use, which might otherwise be impossible given the low input power and high power consumption of a USB charge-only platform with large battery capacity.

Platform requirements and motivation

Article • 02/11/2020

The Windows hardware requirements for the power and battery subsystem are motivated by the need to meet user expectations and to provide fault-tolerant platform operation.

> **Note**
>
> Systems that can only charge from a Universal Serial Bus (USB) port might not be able to support all of the requirements that are listed in this article.

Charging occurs when connected to the charger

Users expect the batteries in their Windows platforms to charge whenever they are connected to an external power source. Therefore, the system must always attempt to charge the battery whenever the platform is connected to AC power or a battery charger regardless of the system power state. This holds true across all of the system power states including active (S0), sleep (S3), hibernate (S4), shutdown (S5), hard off (G2/G3), or modern standby. Charging can stop after the battery is fully charged or if a fault condition occurs.

Though not ideal, a system can be designed to charge the battery at a reduced rate when Windows or the firmware has not been booted or is not running. For example, the battery might charge at a slower rate when the system is completely off and connected to the charger. However, this system should perform more rapid charging after it is booted and the ACPI firmware is available to periodically monitor the battery level.

Finally, some systems can charge the battery at a lower rate when the system is in a thermal condition. In this case, heat is reduced by slowing or eliminating battery charging altogether. Thermal conditions should be the exception case in any good system design.

Windows can always boot when connected to AC power

Users expect that they can immediately boot and use their system whenever it is connected to AC power. Therefore, the system must always be able to boot and be fully useable when connected to AC power. This is true regardless of battery charge level, battery/charger state, or whether the battery is present in platforms that have removable batteries.

If the platform requires a minimum battery capacity to boot the firmware and operating system, the platform must ensure that it always reserves the required capacity. The reserved battery capacity must not be exposed to Windows, and the platform must autonomously manage this capacity.

Keeping such a reserve limits the chance that a user encounters a situation in which the battery is completely depleted. However, even when the platform is properly designed to maintain the required reserve level in the battery, the user can still encounter a fully depleted battery. For example, the user might run the battery to the depletion point and then neglect to connect it to the charger for a sufficiently long period of time. As a result, the battery falls below the reserve level due to normal degradation. In this scenario, the platform must delay booting Windows until sufficient charge is added to the battery.

Charging is managed autonomously by the hardware

Because users expect their PC to charge when it is connected to AC power, battery charging must be performed by the system hardware without requiring intervention by firmware, Windows, drivers, or other software running on the main CPU(s). That's because one or more of these components might not be operational or might be in a fault state at any given time.

Charging stops automatically when fully charged or when a fault occurs

The system automatically stops charging when the battery is completely charged or if a fault has occurred. This must be done without requiring intervention by firmware, Windows, drivers, or other software running on the main CPU(s). In addition, the hardware must comply with all battery safety regulations.

Windows does not automatically boot when power is applied

The system remains in the off (ACPI S5/G2) state unless it is manually powered on by the user. After the system is shut down or turned off, it must remain in that state. Therefore, the system does not automatically boot or otherwise visibly turn on when power is applied to the system.

As an option, the system can have a firmware setting to configure the system to automatically turn on when power is applied, provided that this is not the default setting. Such an option is useful in automated testing and lab environments. For more information about this setting, see Automated lab testing considerations.

Power and battery subsystem checklist

Article • 12/05/2019

System designers can use the checklists included in this section to verify that their platform designs and system firmware follow the Windows guidelines for power and battery subsystem operation. Each item in a checklist specifies whether the requirement applies to PCs that implement the traditional ACPI S3 and S4 power states, and/or to PCs that implement the modern standby power model.

In this section

Topic	Description
Power and battery user experience requirements	The checklist in this article contains requirements that the power supply and battery subsystem must meet to achieve a satisfactory overall user experience.
ACPI firmware implementation requirements	The checklists in this article contain requirements that the ACPI platform firmware must meet to properly support the power and battery subsystem.
Platform hardware implementation requirements	Windows supports two hardware topologies for the power and charging subsystem. For S3/S4-based systems, only embedded controller-based solutions are allowed. For modern standby systems, the system designer can choose between an embedded controller-based solution and an SPB-based solution. This topic describes the requirements for each of the two topologies. A system must comply with the requirements for the selected topology only.

Power and battery user experience requirements

Article • 12/05/2019

The checklist in this article contains requirements that the power supply and battery subsystem must meet to achieve a satisfactory overall user experience.

For each requirement in the checklist, the column marked "Applies to S3/S4" indicates whether the requirement applies to PCs that implement the traditional ACPI S3 and S4 power states. The column marked "Applies to CS" indicates whether the requirement applies to PCs that implement the modern standby power model, which is supported starting with Windows 8. Any exceptions to these requirements are noted.

Requirement	Applies to S3/S4	Applies to MS
Charging occurs when the platform is connected to a charger. This applies to all supported system power states.		
• On (ACPI S0) / Active	Yes	Yes
• Sleep (ACPI S3)	Yes	No
• Hibernate (ACPI S4)	Yes	Yes (x86 only)
• Shutdown (ACPI S5/G2)	Yes	Yes
• Modern standby	No	Yes
Windows can boot when a charger is connected.		

Requirement	Applies to S3/S4	Applies to MS
• The system can be immediately booted from Shutdown (ACPI S5/G2) by the end user pressing the power button immediately after the charger is connected.	Yes	Yes
• The system can be immediately booted regardless of battery charge level, battery/charger state, or whether the battery is present in a system that has a battery that can be removed by the end user.	Yes	Yes
• If the platform requires a minimum battery capacity to always boot, this reserve capacity must be completely managed by the platform and must not be exposed to Windows.	Yes	Yes
• • Under no circumstance should the system automatically boot from the Hibernate (ACPI S4) state or the Shutdown (ACPI S5) state.	Yes	Yes (Except for platforms that have Universal Serial Bus (USB) charging input only.)
Battery charging is autonomously managed by the hardware.		

Requirement	Applies to S3/S4	Applies to MS
• Charging the battery does not require code to run on the main CPUs in form of UEFI firmware, ACPI firmware, or a Windows device driver. The rate of charge may increase when Windows is booted and ACPI firmware is enumerated.	Yes	Yes (Except for platforms that have USB charging input only.)
Charging stops automatically when fully charged or when a fault occurs.		
• The system hardware must automatically stop charging the battery after the battery is fully charged or when a fault occurs with the battery pack.	Yes	Yes
• The system must not automatically power on the main CPUs when power is attached and the system is in a Shutdown (ACPI S5) state.	Yes	Yes
• The battery and charging subsystem must comply with all required regulatory standards for the region of sale for the final system.	Yes	Yes
Real-Time Clock (RTC) backup		

Requirement	Applies to S3/S4	Applies to MS
• A battery backup solution is provided to power the RTC for a minimum of four weeks. This solution is implemented either by having a dedicated RTC backup battery, or by maintaining the required reserve level in the main system battery.	Yes	Yes
Prevent system startup when battery is completely depleted (DC only).		
• System firmware prevents the system from booting Windows if the system battery is below the critical battery threshold (as specified by _BIX.DesignCapacityofLow) and the device is not connected to AC power or a battery charger.	Yes	Yes
• When a low-battery condition prevents Windows from booting, a visual notification must be displayed. This notification uses the "Battery Discharged Completely" glyph. This requirement is waived in cases in which the battery is so depleted that there is insufficient capacity to turn on the display.	Yes	Yes
(Optional) Add an LED to indicate that power is present.		

Requirement	Applies to S3/S4	Applies to MS
• The preferred location for this LED is on the power adapter or the power connector.	Yes	Yes
• The LED can optionally indicate charge status.	Yes	Yes
• The LED must not vary in light intensity or otherwise blink. Color changes are allowed to indicate status changes.	Yes	Yes

ACPI firmware implementation requirements

Article • 12/05/2019

The checklists in this article contain requirements that the ACPI platform firmware must meet to properly support the power and battery subsystem. For each requirement in a checklist, the column marked "Applies to S3/S4" indicates whether the requirement applies to PCs that implement the traditional ACPI S3 and S4 power states. The column marked "Applies to CS" indicates whether the requirement applies to PCs that implement the modern standby power model, which is supported starting with Windows 8.

Power supply checklist

Requirement	Applies to S3/S4	Applies to CS
Power Source device object is present in the ACPI namespace.		
• Device() object for an AC adapter/Power Source is present in the ACPI namespace.	Yes	Yes
Power source device provides the following control methods and objects:		
• Hardware ID (_HID) with a value of "ACPI0003".	Yes	Yes

Requirement	Applies to S3/S4	Applies to CS
• Power Source (_PSR) control method to report whether the power source is currently online (AC power) or offline (battery power). All input power sources for the system must be multiplexed through this _PSR method. For example, _PSR must indicate that the power source is online if the system is powered through a DC barrel connector or a separate dock connector. Do not use multiple ACPI Power Source Devices.	Yes	Yes

Battery checklist

System designers should make sure that they have completed the items in the following table for each battery present in the system. For systems that have multiple batteries, the first battery in the namespace should be the primary battery for the system, for debugging purposes.

Requirement	Applies to S3/S4	Applies to CS
A Device() object for each battery device in the ACPI namespace.		
• Device() object for each battery is present in the ACPI namespace.	Yes	Yes
Each battery device provides the following control methods and objects:		

Requirement	Applies to S3/S4	Applies to CS
• _HID with a value of "PNP0C0A".	Yes	Yes
• General Status (_STA) to report to Windows whether the battery is present in a system in which the battery can be removed or the battery is located in a portable dock.	Yes	Yes
• Battery Information Extended (_BIX) to report battery static information, which includes last full charge capacity, design capacity, and cycle count.	Yes	Yes
• Battery Status (_BST) to report the current battery status, including remaining capacity, rate of drain and charging state.	Yes	Yes
• Battery Trip Point (_BTP) to enable an event-driven battery status model to reduce periodic work for polling. _BTP enables Windows to specify a threshold of remaining charge capacity at which the platform should issue a Notify(0x80) command on the battery device to notify Windows that it needs to update its battery status information.	Yes	Yes

Requirement	Applies to S3/S4	Applies to CS
• (Optional) Slot Unit Number (_SUN) to indicate the order in which the battery should appear in the UI.	Yes	Yes
The _BIX method must support the following fields and constraints as described in battery static information section:		
• The Revision field must be set to 0x0.	Yes	Yes
• The Power Unit field must be set of 0x0.	Yes	Yes
• The Design Capacity and Last Full Charge Capacity values must be set to accurate values from the battery and charging subsystem and not set equal to 0xFFFFFFFF or 0x00000000.	Yes	Yes
• The Battery Technology field must be set to 0x1.	Yes	Yes
• The Design Voltage field must be set accurately and not set equal to 0x00000000 or 0xFFFFFFFF.	Yes	Yes

Requirement	Applies to S3/S4	Applies to CS
• The Design Capacity of Low must be set to the minimum value required to Hibernate or Shutdown the system from a fully on state.	Yes	Yes
• The Battery Capacity Granularity 1 and Battery Capacity Granularity 2 fields must be set to a value no larger than 1 percent of the battery design capacity.	Yes	Yes
• The Cycle Count field must be accurately filled in from the battery subsystem.	Yes	Yes
• The Measurement Accuracy field must be set to 80,000 or better.	Yes	Yes
• The Model Number and Serial Number fields must not be set to NULL.	Yes	Yes
• When information in _BIX, including last full charge capacity, design capacity, and cycle count changes, the platform will issue an ACPI Notify(0x81) command on the battery device.	Yes	Yes
The _BST method must report real-time battery status.		

Requirement	Applies to S3/S4	Applies to CS
• All information returned by the _BST method must be obtained dynamically from the underlying power and battery charging subsystem.	Yes	Yes
• Data accuracy must be within the value of **Measurement Accuracy** as reported in the _BIX object.	Yes	Yes
Support for ACPI Battery Trip Points		
• A _BTP method is provided that allows Windows to specify a remaining charge capacity threshold at which the platform will issue an ACPI Notify(0x80) command on the battery device.	Yes	Yes
• The battery device issues an ACPI Notify command when the battery charge reaches the critical level.	Yes	Yes
• When the battery charge level reaches the value specified in _BIX.DesignCapacityofLow, the platform must generate a Notify(0x80) command on the Control Method Battery device.	Yes	Yes

Requirement	Applies to S3/S4	Applies to CS
Implement Device Specific Methods as appropriate		
• Implement the Device Specific Method (_DSM) under each battery device to indicate if the battery is user-serviceable.	Yes	Yes
• Implement the _DSM method if a periodic watchdog reset is required during charging and Windows will guarantee execution of the _BST method within the polling window for each period.	Yes	Yes
• Implement the _DSM method if the platform's thermal model requires control over the battery charging rate.	Yes	Yes

Platform hardware implementation requirements

Article • 12/05/2019

Windows supports two hardware topologies for the power and charging subsystem. For S3/S4-based systems, only embedded controller-based solutions are allowed. For modern standby systems, the system designer can choose between an embedded controller-based solution and an SPB-based solution. This topic describes the requirements for each of the two topologies. A system must comply with the requirements for the selected topology only.

Each of the following two tables contains a checklist of requirements for the designated hardware topology. For each requirement in the checklist, the column marked "Applies to S3/S4" indicates whether the requirement applies to PCs that implement the traditional ACPI S3 and S4 power states. The column marked "Applies to CS" indicates whether the requirement applies to PCs that implement the modern standby power model, which is supported starting with Windows 8.

For more information about these two hardware topologies, see Battery and power subsystem hardware design.

Embedded controller-based solution

If the platform designer chooses to use an embedded-controller based solution, the requirements that are listed in the following table apply.

Requirement	Applies to S3/S4	Applies to CS
The embedded controller (EC) is defined in the ACPI namespace as described in section 12.11 of the ACPI 5.0 specification.		
• Device() object for the EC is present in the ACPI namespace.	Yes	Yes
EC device provides the following control methods and objects:		

Requirement	Applies to S3/S4	Applies to CS
• _HID with a value of "PNP0C09".	Yes	Yes
• _CRS object denoting the I/O resources for the embedded controller.	Yes	Yes
• _GPE object that defines the SCI for the embedded controller.	Yes	Yes
• An operation region describing the information contained within the embedded controller that can be accessed by other ACPI control method code in the namespace, including battery status and information methods.	Yes	Yes
Embedded controller solution meets power goals.		
• Nominal power consumed by the EC for power and battery subsystem is below 5 milliwatts.	No	Yes

Simple peripheral bus (SPB)-connected solution

If the platform designer chooses to use an SPB-connected power and battery subsystem solution, the requirements that are listed in the following table apply. This solution is supported starting with Windows 8, but only on modern standby systems.

Requirement	Applies to S3/S4	Applies to CS
The embedded controller (EC) is defined in the ACPI namespace as described in section 12.11 of the ACPI 5.0 specification.		
• Device() object for the EC is present in the ACPI namespace.	No	Yes
EC device provides the following control methods and objects:		
• _HID with a value of "PNP0C09".	No	Yes
• _CRS object denoting the I/O resources for the GPIO controller.	No	Yes
• For each interrupt, an _AEI object that maps the GPIO to the corresponding ACPI event method.	No	Yes
A Device() node for the SPB controller.		
• _HID with the appropriate hardware ID for the device.	No	Yes
• _CSR object describing the interrupt and hardware resources of the GPIO controller.	No	No

Requirement	Applies to S3/S4	Applies to CS
• A GenericSerialBus operation region within the scope of the SPB device describing the virtual command registers for the SPB device.	No	Yes
• Field definitions within the GenericSerialBus operation region. The field definitions allow ASL code outside of the SPB device to access the virtual command registers for the SPB device.	No	Yes

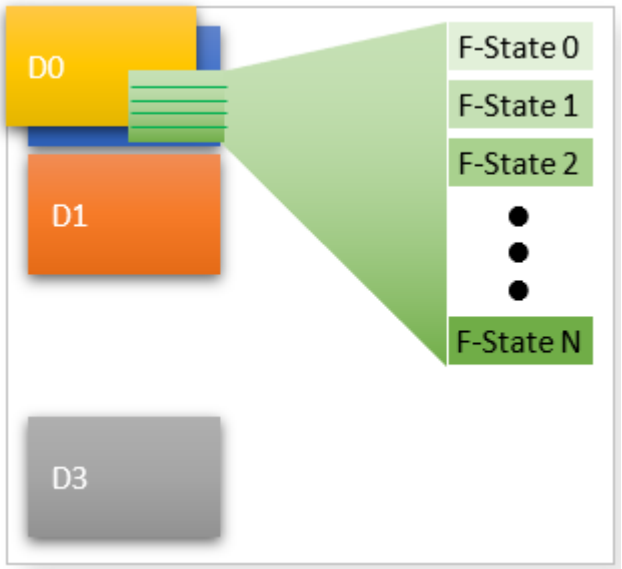
Power Management for Storage Hardware Devices

Article • 03/22/2021

Introduction

This document is intended as a guide for OEMs and device manufacturers using Microsoft's in-box SATA and NVMe drivers on Windows 10 May 2019 Update or newer OS versions. Microsoft's general guidance is to provide in-box solutions that "just work" in terms of balancing performance, responsiveness, and power efficiency. As is the case with overall system performance, poorly-behaved device hardware may compromise overall efficiency and battery life. In general, the system attempts to balance power efficiency with performance.

The diagram below illustrates ACPI-defined device power states D0 (working) to D3 (off). Note that as the device transitions into deeper power states, power consumption is reduced, but latency increases. Also note that Windows supports varying functional states (F-States), which can be used to control fine-grained levels functionality and power within the D0 state. This can be useful for runtime power management where the system needs to remain highly responsive, but still needs to save power.



ACPI-Defined Sleep State Power Management

When the system is not in use, Windows may opportunistically place the system in an ACPI-defined sleep state to conserve energy. Likewise, Windows may choose deep sleep states over time to conserve even more power. For example, the system may transition

to S3 for a period, and eventually transition to S4 Hibernate. When the system transitions to a sleep state, the general rule of thumb is to place the device in the deepest possible D-state, unless the device is wake-capable, and is armed for wake. Under these conditions, a shallower D-state may be appropriate. Likewise, when the system wakes, the device will transition back to D0.

Runtime Power Management

In order to achieve maximum power efficiency, some components will implement very fine-grained idle logic to determine when devices can be powered down, even when the system is in active use. E.g., a high-end storage device could disable certain functional blocks during runtime if the driver deems that they haven't been used for some period of time. This is only possible if those functional blocks can be brought back online and made functional quickly enough such that the user does not incur noticeable latencies.

Modern Standby Power Management

When the system is not in use, Windows may opportunistically turn off power to some set of devices to conserve energy. In Modern Standby, the system remains in S0. Even while in S0, all peripheral devices may eventually be powered down due to idle timeouts. This state is defined as "S0 Low Power Idle". Once all devices are in a low-power state, even more of the system infrastructure (e.g. busses, timers, …) may be powered down. The general rule of thumb is to place the device in the deepest possible D-state when it is idle, even when the system state is S0. Depending on implementation details of the processor complex and platform design, peripheral devices may be required to go to an F-state, D3 Hot, or D3 Cold (power is cut). To mitigate the need for a function driver to manage these implementation details, drivers should go to the deepest appropriate device state in order to maximize battery life.

D3 Support

When the system is not in use, Windows may opportunistically turn off power to some set of devices to conserve energy. In Modern Standby, the system remains in S0. Even while in S0, all peripheral devices may eventually be powered down due to idle timeouts. This state is defined as "S0 Low Power Idle". Once all devices are in a low-power state, even more of the system infrastructure (e.g. busses, timers, …) may be powered down. The general rule of thumb is to place the device in the deepest possible D-state when it is idle, even when the system state is S0. Depending on implementation details of the processor complex and platform design, peripheral devices may be required to go to an F-state, D3 Hot, or D3 Cold (power is cut). To mitigate the need for

a function driver to manage these implementation details, drivers should go to the deepest appropriate device state in order to maximize battery life.

```
ASL Copy
Name (_DSD, Package () {

          ToUUID("5025030F-842F-4AB4-A561-99A5189762D0"),

            Package () {

                Package (2) {"StorageD3Enable", 1}, // 1 - Enable; 0 -
Disable

            }
        }
 )
```

The ACPI _DSD above is the preferred way to opt into or out of D3 support for storage devices. However, there is also a global registry key that can be used to modify D3 support if needed.

- Name: StorageD3InModernStandby
- Type: REG_DWORD
- Path: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Storage\
- Value:
  - 0 – Disable D3 support
  - 1 – Enable D3 support

If the registry key is not configured, then Storport will check the _DSD configuration to determine whether to enable D3. If the _DSD is not implemented, then Storport will check whether the platform is on the allowlist for D3 support.

Parent/Child Relationship for Power Management

During power-up, the parent/child relationship is always enforced for storage devices. During power-down, the only case where the parent/child power relationship is not enforced is if the controller only supports D3Hot, and the device reports F1 to PoFx (i.e. DEVSLP is supported or it is an SSD in a Modern Standby system), then the controller is allowed to enter D3 while the device is in F1.

In this section

Topic	Description
NVMe	This topic covers power management guidelines for NVMe storage devices.
SATA/AHCI	This topic covers power management guidelines for SATA/AHCI storage devices.

NVMe

Article • 06/12/2023

This section details how Microsoft's in-box NVMe driver (StorNVMe) manages power and what configuration options are available. The NVMe spec allows NVMe devices to report up to 32 power states. Each power state has the following parameters:

- Maximum Power Consumption (MP)
- Operational or Non-operational
- Entry Latency (ENLAT)
- Exit Latency (EXLAT)
- Relative performance values (relative to other power states)

StorNVMe maps operational power states (the device can handle IO in these states) to logical performance states (a.k.a. P-States). Likewise, the driver maps non-operational power states (the device does not handle IO in these states) to logical idle power states (a.k.a. F-States). With StorNVMe, transitions to these states is largely determined by the overall system power state. The NVMe specification defines an Autonomous Power State Transition (APST) feature. For Modern Standby support, StorNVMe does not support devices with APST enabled.

Runtime Device Power Management

StorNVMe may choose to transition the device to an F-State after a certain amount of idle time has elapsed. The F-State is chosen based on 3 factors:

1. **Latency Tolerance**, which is how quickly the device can respond if needed. For F1, the TransitionLatency (ENLAT + EXLAT) shouldn't be larger than Primary Transition Latency Tolerance. For F2 and other deeper F-states (if any), their TransitionLatency shouldn't be larger than Secondary Transition Latency Tolerance. Otherwise, the device might not be able to transition to these F-states, and modern standby transition might be impacted (for instance, introduce long latency to enter DRIPS).
2. **Idle Timeout**. This is the amount of time from when the device completed its last IO operation.
3. **System Power State**. If the system is in active use, StorNVMe will prefer responsiveness. This implies varying latency tolerances and timeouts will be used.

The table shows the default idle timeouts and latency tolerances used by StorNVMe. See the Power Configuration Settings section for how to change these settings.

ACPI System Power State	Primary Idle Timeout	Primary Transition Latency Tolerance	Secondary Idle Timeout	Secondary Transition Latency Tolerance
S0 (Working) - Performance Scheme	200ms	0ms (AC) / 10ms (DC)	2000ms	0ms
S0 (Working) - Balanced Scheme	200ms (AC) / 100ms (DC)	15ms (AC) / 50ms (DC)	2000ms (AC) / 1000ms (DC)	100ms
S0 (Working) - Power Saver Scheme	100ms	100ms (AC) / 200ms (DC)	1000ms	200ms
S0 Low Power Idle (Modern Standby)	50ms	500ms	N/A	N/A

Once the idle timeout has expired, the driver will traverse its internal table of power states and select the deepest power state where ENLAT+EXLAT is less than or equal to the current transition latency tolerance.

For example, assume an NVMe device has the following power states, and that an idle timeout has occurred:

Power State	Entry Latency (ENLAT)	Exit Latency (EXLAT)
PS0	5us	5us
PS1	10ms	300us
PS2	50ms	10ms

When the system is on DC power and not in Modern Standby, StorNVMe will choose PS1 since this is the deepest power state where (ENLAT+EXLAT) <= 50ms. Likewise, when the system enters Modern Standby, StorNVMe will then choose PS2 because it is the deepest power state where (ENLAT+EXLAT) <= 500ms.

Modern Standby and DRIPS

To fully support Modern Standby, StorNVMe will transition the device to an appropriate low-power state depending on hints provided by the hardware platform. The idle state will vary between an F-State (deeper than F0), to D3 Cold. Some platforms require D3 Cold when in Modern Standby. This depends on the SoC, so please check with your

silicon vendor for more information. D3 support for storage devices on Modern Standby systems can be enabled as described here.

Devices should support RTD3 with short resume latency in order to help Modern Standby systems meet the 1 second system resume latency requirement. RTD3 Resume Latency (RTD3R) refers to resume latency from D3cold and is recommended to report a nonzero value ≤ 100 ms. **RTD3R** is described in section 8.4.4 of the NVMe spec.

Power Configuration Settings

Windows 10 supports the following NVMe power settings to tune energy efficiency.

Primary NVMe Idle Timeout

The following power configuration setting allows you to change the primary device idle timeout used by StorNVMe.

```
Power Setting GUID: d639518a-e56d-4345-8af2-b9f32fb26109  (Primary NVMe Idle
Timeout)
      Minimum Possible Setting: 0x00000000
      Maximum Possible Setting: 0x0000ea60
      Possible Settings increment: 0x00000001
      Possible Settings units: milliseconds
```

The following power configuration setting allows you to change the primary transition latency tolerance value that StorNVMe uses when calculating an idle state. This is the value that is compared against the sum of the ENLAT and EXLAT values when the idle timeout expires. **The higher this value, the more likely that a deeper power state will be chosen.**

```
Power Setting GUID: fc95af4d-40e7-4b6d-835a-56d131dbc80e  (Primary NVMe
Power State Transition Latency Tolerance)
      Minimum Possible Setting: 0x00000000
      Maximum Possible Setting: 0x0000ea60
      Possible Settings increment: 0x00000001
      Possible Settings units: milliseconds
```

Secondary NVMe Idle Timeout

The following power configuration setting allows you to change the secondary device idle timeout used by StorNVMe.

```
Power Setting GUID: d3d55efd-c1ff-424e-9dc3-441be7833010  (Secondary NVMe
Idle Timeout)
      Minimum Possible Setting: 0x00000000
      Maximum Possible Setting: 0x0000ea60
      Possible Settings increment: 0x00000001
      Possible Settings units: milliseconds
```

The following power configuration setting allows you to change the secondary transition latency tolerance value that StorNVMe uses when calculating an idle state. This is the value that is compared against the sum of the ENLAT and EXLAT values when the idle timeout expires. **The higher this value, the more likely that a deeper power state will be chosen.**

```
Power Setting GUID: dbc9e238-6de9-49e3-92cd-8c2b4946b472  (Secondary NVMe
Power State Transition Latency Tolerance)
      Minimum Possible Setting: 0x00000000
      Maximum Possible Setting: 0x0000ea60
      Possible Settings increment: 0x00000001
      Possible Settings units: milliseconds
```

To change the value for a given power scheme, use:

Windows Command Prompt

```
powercfg [-setacvalueindex | -setdcvalueindex] <scheme> sub_disk <Power
Setting GUID> <milliseconds>
```

Don't forget to apply the value by using: `powercfg –setactive <scheme>`

PCIe ASPM and L1 Sub-states

Depending on your platform, you may observe that the NVMe device is able to enter L1 sub-states when on DC power but not on AC power. In this case, you may need to change the PCIe ASPM power configuration setting such that it gets maximum power savings when on AC power (in addition to DC power).

```
Power Setting GUID: ee12f906-d277-404b-b6da-e5fa1a576df5  (Link State Power
Management)
      GUID Alias: ASPM
      Possible Setting Index: 000
      Possible Setting Friendly Name: Off
      Possible Setting Index: 001
      Possible Setting Friendly Name: Moderate power savings
      Possible Setting Index: 002
      Possible Setting Friendly Name: Maximum power savings
```

To change the value, use:

`powercfg -setacvalueindex <scheme> sub_pciexpress aspm <value>`

with Index 002 from above for maximum power savings. Don't forget to apply the value by using: `powercfg –setactive <scheme>`

Active Power Management

Active power management involves "P-States" (a.k.a. performance or "perf" states) and is primarily intended for thermal control. StorNVMe maps the device's operational power states to logical P-States using the Maximum Power value reported for each operational power state. When the device is active (i.e. has outstanding IO) then StorNVMe will transition the device to one of its operational power states via a P-State transition.

During Windows 10 development there was a limited set of NVMe devices that implemented more than one operational power state. Based on our power and performance measurements, we did not find a significant benefit to using any but the highest operational power state. Therefore, with the default configuration you will only see the highest operational power state utilized.

The operational power state chosen depends upon the current "max operational power" hint. This hint can have 3 different sources:

- A passive cooling callback from the Windows Thermal Framework.
- A change in the maximum power level power configuration setting value. (This can be triggered by a change in system power scheme or AC/DC power source.)
- A IOCTL_STORAGE_DEVICE_POWER_CAP request. The lowest maximum value from these sources is the effective maximum operational power value. The mechanics of each of these sources is discussed below.

In general, StorNVMe will choose the highest operational power state that is less than or equal to the effective maximum operational power value.

For example, say an NVMe device has the following power states:

Power State	Max Power	Operational?
PS0	9W	Yes
PS1	6W	Yes
PS2	4W	Yes

The lowest maximum value from these sources is the effective maximum operational power value. The mechanics of each of these sources is discussed below.

In general, StorNVMe will choose the highest operational power state that is less than or equal to the effective maximum operational power value.

By default, there is no maximum power level so StorNVMe will always choose PS0. This is equivalent to 100%.

If the Windows Thermal Framework calls the passive cooling callback with a value of 50% then this will result in an absolute power value of (50% * (9W – 4W)) + 4W = 6W. StorNVMe will then ensure that when the device is active it will always be in PS1 since that state's Max Power value is 6W.

Then some user mode process sends an IOCTL_STORAGE_DEVICE_POWER_CAP request to the disk with a value of 5W. StorNVMe will choose PS2 now because it is the highest operational power state whose Max Power value (4W) is less than the max operational power requirement of 5W.

If the given max operational power requirement is less than the Max Power value of the lowest operational power state, then the expectation is to simply choose the lowest operational power state. In our example, if the max operational power requirement given was 3W then StorNVMe would choose PS2 because it has no operational power state with a Max Power value of 3W or less.

If later the max operational power requirement changes to 9W then StorNVMe will go back to choosing PS0 when the device is active.

For example, say an NVMe device has the following power states:

Windows Thermal Framework Passive Cooling Callback

StorNVMe (via Storport) registers a thermal cooling interface with the Windows Thermal Framework which makes it possible for the system to throttle the NVMe device through that framework. The specifics of this are outside the scope of this document, but in general the platform specifies thermal zones and thresholds via ACPI which the Windows Thermal Framework then uses to throttle devices via callbacks to the devices' drivers.

Maximum Operational Power Level Power Configuration Setting

The following power configuration setting can be used to change the maximum operational power level for different system power schemes and AC/DC power source.

```
Power Setting GUID: 51dea550-bb38-4bc4-991b-eacf37be5ec8  (Maximum Power
Level)
      GUID Alias: DISKMAXPOWER
      Minimum Possible Setting: 0x00000000
      Maximum Possible Setting: 0x00000064
      Possible Settings increment: 0x00000001
      Possible Settings units: %
```

To change the value for a given power scheme, use:

Windows Command Prompt

```
powercfg [-setacvalueindex | -setdcvalueindex] <scheme> sub_disk 51dea550-
bb38-4bc4-991b-eacf37be5ec8 <value>
```

Don't forget to apply the value by using: `powercfg –setactive <scheme>`

IOCTL_STORAGE_DEVICE_POWER_CAP

This IOCTL can be sent to a storage device to change the maximum operational power level. For more info, see the documentation for the input/output buffer, STORAGE_DEVICE_POWER_CAP.

Shutdown/Hibernate

When the system is being shut down or hibernated, StorNVMe sets the device's Shutdown Notification (CC.SHN) field to 1. StorNVMe then waits for the device's

reported RTD3 entry latency for the device to indicate it is ready (by updating the Shutdown Status (CSTS.SHST) field to 2). If no entry latency value is reported, then StorNVMe will use a default value of 5 seconds. In this case, if the device takes longer than 5 seconds, then the system will continue with the shutdown or hibernate procedure without checking the NVMe device any further. OEMs should only use devices that report RTD3 entry and exit values for Modern Standby systems.

SATA/AHCI

Article • 08/24/2021

This section details how Microsoft's in-box SATA driver (StorAHCI) manages power and what configuration options are available.

HIPM/DIPM

By default, HIPM-only is enabled and StorAHCI manages Partial to Slumber transitions. When the device is in D0, after the link has been in Partial for 100ms, StorAHCI will transition to Slumber. StorAHCI allows the controller to handle transitions from Active to Partial by enabling the "Aggressive Partial feature" of the SATA-I/O standard.

After the device enters D3, StorAHCI will immediately transition the link to Slumber.

HIPM-only was chosen because StorAHCI controls DEVSLP transitions directly and typically DEVSLP-capable controllers indicate that DEVSLP can only be reached through the Slumber state. Thus, StorAHCI also needs to control transitions to Slumber.

DEVSLP

StorAHCI controls DEVSLP directly in order to effectively balance power, responsiveness, and diagnosability. Thus, StorAHCI does not use the hardware-controlled DEVSLP feature (a.k.a. "Aggressive DEVSLP" per the SATA-I/O spec.).
DEVSLP is mapped to a single logical idle power state or "F-State," namely F1.

The following table shows the time the SATA device must be idle before it transitions to DEVSLP under different system power states. Note that if the controller has specified that DEVSLP must be entered from Slumber, StorAHCI will first transition to Slumber and then upon completing the transition to Slumber it will immediately transition to DEVSLP. As mentioned above, this implies that HIPM must be supported.

System Power State	DEVSLP Idle Timeout
S0 (Working)	6 seconds
S0 Low Power Idle (Modern Standby (MS))	50ms

Adaptive D3 Idle Timeout

Now that SATA drives with rotational media are supported on Modern Standby systems, there is a need to balance power savings with device reliability. The device needs to be powered down more aggressively when in Modern Standby in order to meet the system's power requirements. However, powering down a rotational drive too aggressively can result in excessive wear on the drive's mechanical parts. To help reduce wear and tear, Windows 10 includes Adaptive D3 Idle Timeout, where the device's power cycle count is tracked and compared against a worst case model based on a typical device warranty (the worst case being where the drive would only last a couple of years). If the actual power cycle frequency is trending too close to the worst case model, then the D3 idle timeout is increased to allow the trend to dip back into safer numbers. If the power cycle frequency is low enough such that the device is not in danger of experiencing excess wear, then the D3 idle timeout value is shortened dramatically to ensure the drive is powered down quickly once it is idle in Modern Standby.

StorAHCI specifies a minimum power cycle period of 5 minutes for rotational drives. This means that if such a drive is being power cycled more frequently than every 5 minutes it will instead remain powered up when idle for a short time to compensate. If you want to modify the minimum power cycle period, according to guidance from your device vendor, you can use the following registry key:

- Name: MinPowerCyclePeriodInSecs
- Type: MULTI_SZ
- Path: HKLM\System\CurrentControlSet\Services\storahci\Parameters\Device
- Value: `<Product ID> <Value>`, e.g. "ST31000528AS 300" or "WDC WD4* 360"
  - To specify a pattern to match multiple product IDs, use:
    - '?' to match any single character
    - '*' to match any remaining characters
  - The value itself is in units of seconds.

Device Maintenance Time

Because it is now possible for rotational drives to spend a significant amount of time powered down, Windows 10 also includes a mechanism that gives the drive some powered-up idle time (1 minute every 24 hours) to perform internal maintenance. This only occurs when the system is connected to AC power in order to preserve battery life and when the system is in Modern Standby in order to ensure disk activity is at a minimum. Device maintenance is not configurable.

Modern Standby and DRIPS

Starting with Windows 10, drives with rotational media (HDDs or SSHDs) are supported on Modern Standby systems. HDDs may result in higher power drain due to Adaptive D3 Idle Timeout keeping the disk in D0 for longer periods. HDDs may also result in longer exit latency from Modern Standby. However, Modern Standby systems with HDDs are exempted from the 1s system resume latency requirement. SSDs, especially for the primary boot drive, are recommended over HDDs if possible.

Regardless of the media type of the storage device(s) on the system, in order to support Modern Standby, the platform needs to specify a constraint on either:

1. Each SATA port; or
2. On the AHCI controller

This constraint is consumed by the Power Engine Plug-in (PEP) and should allow the system to enter its Deepest Runtime Idle Power State (DRIPS) when either:

1. All SATA drives enter DEVSLP (F1) or deeper (D3 is considered deeper than F1); or
2. The AHCI controller enters the F1 state or deeper.

The specifics of this are platform-specific and outside the scope of this document.

> ⓘ **Note**
>
> Microsoft does not recommend that SATA SSDs/HDDs support Automatic Partial to Slumber Transitions (APST). APST will be automatically disabled.

PCIe-Connected AHCI SSDs

During Windows 10 development none of the PCIe-connected AHCI SSDs we encountered exposed a DEVSLP state to StorAHCI. This means Windows 10 has virtually no options when it comes to managing power for these devices. In this case, the device and platform bear the bulk of the power management responsibility.

Bluetooth

Article • 11/18/2022

Features and Recommendations

This is a list of features and recommendations for Bluetooth in Windows. For power-efficient support for Bluetooth LE beacons and related scenarios, Microsoft recommends Bluetooth components capable of hardware offload and able to support the Microsoft-defined Host Controller Interface (HCI) extension.

Additional features that were added after Windows 10, version 1507 are marked with their corresponding release number.

QD IDs

Expand table

QD ID	Name	Product type	Date
192465	Windows 11, version 22H2	Host Subsystem	06-June-2023
165530	Windows 11, version 22H2	Host Subsystem	04-October-2022
165530	Windows 11, version 21H2	Host Subsystem	07-July-2021
165530	Windows 11	Host Subsystem	07-July-2021
130847	Windows 10, version 22H2	Host Subsystem	18-November-2022
130847	Windows 10, version 21H2	Host Subsystem	26-October-2021
130847	Windows 10, version 21H1	Host Subsystem	07-July-2021
130847	Windows Server 2022	Host Subsystem	26-May-2021
130847	Windows 10, version 20H2	Host Subsystem	26-January-2021

QD ID	Name	Product type	Date
130847	Windows 10, version 2004	Host Subsystem	10-January-2020
121270	Windows 10, version 1909	Host Subsystem	19-November-2019
121270	Windows 10, version 1903	Host Subsystem	29-March-2019
108589	Windows 10, version 1809	Host Subsystem	07-September-2018
106476	Windows 10, version 1803	Host Subsystem	10-April-2018
94254	Windows 10, version 1703	Host Subsystem	14-March-2017
84637	Windows 10 Mobile updated AVRCP	Profile Subsystem	17-June-2016
70587	Windows 10 for Desktop editions (Home, Pro, Enterprise, and Education)	Host Subsystem	14-July-2015
72550	Windows 10 Mobile	Profile Subsystem	31-July-2015

Bluetooth features

- Converged Bluetooth core stack. Audio profiles differ in behavior between Windows 10 for Desktop editions (Home, Pro, Enterprise, and Education) and Windows 10 Mobile.
  - Compliant to the standard Bluetooth 4.1 (for all mandatory spec requirements, not for optional requirements)
  - Support for the following features:
  - Hardware Offload (HCI): Microsoft defines vendor-specific HCI commands and events that are consumed by Windows
  - Dual mode support (except in Windows 10 Mobile): Dual mode support for Bluetooth BR/EDR and Bluetooth LE. For example, a Jabra Sport Pulse can transfer heart rate data over Bluetooth LE and stream music over a BR/EDR profile. Bluetooth BR/EDR, the older standard, is still needed for higher data-rate or long-range applications.
  - LE privacy 1.1: Windows protects the user's privacy by randomizing the Bluetooth address whenever it is broadcast.

    - Enterprise Management
        - Turn Bluetooth on/off
        - Allow/Block the device going into discoverable mode
        - Change the friendly name of the Bluetooth device
        - Allow/Block advertisements
        - (1511) Allow/Block connections and pairings based off of the profiles supported. For example, block file transfers while allowing keyboards and mice.
    - Windows Phone General Distribution Release (GDR) features ported to Windows 10 (Cortana address book entry, LE HID, MAP)
    - (1607) Pre-pair Bluetooth devices during manufacturing: Allows for a seamless out-of-box experience that allows Bluetooth devices to just work.
    - (1703) Hands-Free Profile (HFP) 1.6 specification with Wideband speech on Windows 10 for Desktop editions.
    - (Windows 11 version 22H2) Bluetooth 5.3
    - (Windows 11 version 22H2) Bluetooth LE Audio

- APIs/Profiles
    - Advertisement API: Support for scanning of Bluetooth LE advertisement packets for beacons and other similar scenarios.
    - Existing Windows Phone Background support converged to Desktop.
    - Bluetooth audio enhancements (Wideband speech, aptX®) - High Definition sound while limiting Lip Sync and latency issues.
    - (1511) In App Pairing: Allows apps to discover, pair, and connect to devices all without leaving the app in question. See this GitHub code sample and DeviceInformationPairing class for more information.
    - (1607) Non-paired support for connections over RFCOMM: Apps can now communicate with Windows over RFCOMM without having to pair the device.
    - (1607) Map apps can now use Cortana's voice for navigation.
    - (1703) Support for Call Control APIs on Windows 10 for Desktop editions.
    - (1703) Support for GATT Server, Bluetooth LE Peripheral role and non-paired support for Bluetooth LE. See our developer post for more details.
    - (1803) Support for Swift-Pair. See Bluetooth Swift Pair

Bluetooth features list by edition

The following table provides a summary of the Bluetooth features in the Windows 10 for Desktop editions and Windows 10 Mobile editions.

Expand table

Feature	Windows 10 for Desktop editions	Windows 10 Mobile
Advertisement API	Yes	Yes
In App Pairing API	Yes	Yes
Non-paired support over RFCOMM	Yes	Yes
Existing Windows Phone Background support converged to Desktop	Yes	Yes
Hardware Offload (HCI)	Yes	Yes
Dual mode, LE Privacy 1.1 (Bluetooth Spec updates)	Yes	Dual mode - Yes (Only while using pairing APIs)*
Enterprise Management	Yes	Yes
Bluetooth audio enhancements – Wideband speech	Yes	Yes
Bluetooth audio enhancements – aptX® audio codec	Yes	Yes
Interop Improvements	Yes	Yes
WP GDR Features ported to Windows 10 – Cortana address book entry	No	Yes
WP GDR Features ported to Windows 10 – LE HID (Mobile)	Yes	Yes
WP GDR Features ported to Windows 10 – Message Access Profile (MAP)	No	Yes
WP GDR Features ported to Windows 10 – AVRCP	Yes (v1.3)	Yes (v1.4)
MAP apps access to Cortana's voice	No	Yes
Pre-pair Bluetooth devices during manufacturing	Yes	No
Background Bluetooth process content	Yes	Yes
Support for Swift Pair	Yes	No

Note

This is not supported via the Settings page and must be implemented programmatically using In App Pairing APIs.

Upgrading from Windows 10

During upgrade, Windows will migrate Bluetooth drivers, applications, and profile packs. There is still the ability to check Windows Update for a more current driver, and install during the upgrade process. If Bluetooth is not working or missing after upgrade, please file feedback in the Feedback Hub with details of what is happening. Users may need to check their hardware manufacturer's or OEM's website to install Windows 10 compatible drivers.

Upgrading from Windows 7, Windows 8, or Windows 8.1

During upgrade, Windows 11 will not migrate existing Bluetooth drivers, applications, and profile packs. During the upgrade process, Windows setup will check Windows Update to download and install a Windows 11 Bluetooth filter driver to re-enable Bluetooth radio functionality with the inbox profile support listed in the table above. Users will have to check their hardware manufacturer's or OEM's website for updated profile packs if desired.

Maintained guidelines from Windows 8.1

This section cover guidelines from Windows 8.1 that are maintained for Windows 10.

Hardware

If implemented, Bluetooth controllers must support the Bluetooth 4.0+LE specification, complying with both Basic Rate (BR) and Low Energy (LE).

The following table summarizes the supported peripheral buses and driver support.

Expand table

Bus (HCI)	Driver support	SCO support
Non-USB	WDK sample	Sideband I2S/PCM connection only (HCI bypass)
USB	In-box	In-band (SCO over HCI)

The WDK Bluetooth Serial HCI Bus Driver sample is based on the UART (H4) standard as defined in the Bluetooth SIG specification. A vendor will be required to adopt and enhance the sample for any vendor-specific device requirements around device initialization and/or power management. If desired, the vendor can adopt the sample and develop for a non-UART interface as well, i.e.: non-UART controllers will also be supported by the Bluetooth stack (given a proper vendor-supplied driver).

A vendor supplied serial controller driver is necessary for UART-based controllers. For UART-specific features, see Simple Peripheral Bus (SPB).

> **Note**
>
> A non-USB connected Bluetooth controller must use a sideband channel for SCO applications, i.e.: SCO over I2S/PCM interface. SCO over HCI (in-band) will not be supported for non-USB controllers.

Transport bus driver

The Windows Driver Kit (WDK) sample is available for the UART (H4) transport. A vendor can enhance it for any vendor-specific feature, including for any non-UART transports as well. There will be no limitations around the stack's ability to support a particular transport.

There will be no changes to the existing in-box Bluetooth USB driver. We recommend using UART (H4) as the connectivity interface, since the WDK sample will be UART-based and due to UART's lower power consumption. Voice (SCO) support must go through a "sideband" audio channel for non-USB controllers, such as an I2S/PCM interface.

Initialization and power handling

For non-USB based Bluetooth controllers that require initialization see Transport Bus Driver for Bluetooth Power Handling Guidelines.

Radio management

The 3rd-party Bluetooth radio management plugin is not supported as Bluetooth Radio Management support is now provided inbox. Transport drivers must respond to being D3 by turning off power to the radio.

Mechanical

We do not recommend an external switch for controlling the on/off state of the Bluetooth radio.

Third-party Bluetooth software

3rd party software can be added to x86/x64 Windows PCs to provide additional Bluetooth profile functionality not natively shipped in Windows. To avoid impacting the Windows user experience, causing incompatibilities with other Windows PCs, and creating serviceability issues on upgrade, Windows recommends the following:

- Make them installable by INF so they can be easily serviced to support Windows as a Service.
- Do not replace inbox profiles, icons, or user interfaces.
- When adding profiles and/or other software, use the native Windows APIs.
- Use Wi-Fi Direct for high bandwidth peer-to-peer scenarios instead of Bluetooth High Speed (HS).
- Optimal app footprint to minimize impact on manufacturing time.
- Optimal app performance to minimize impact on off to on transitions (boot, resume from S3/S4) and energy efficiency.

Related topics

- Advertising API
- Background support
- GattCharacteristicNotificationTrigger
- RfcommConnectionTrigger
- Hardware Offload (HCI)
- Dual mode, LE Privacy 1.1 (Bluetooth Core Spec updates)
- Bluetooth audio enhancements - aptX® audio codec

Bluetooth LE pre-pairing

Article • 03/17/2021

Windows 10 supports LE Bluetooth peripherals automatically pairing to a single PC, after all devices have been provisioned and enabled during the manufacturing process. Users do not need to keep LE devices paired and connected after both the host PC and bundled devices are supplied power for the first time. These peripherals can still be used as regular Bluetooth devices, and maintain full functionality when out of range of the host PC. However, if the peripherals are unpaired and within range of the host PC, they will automatically re-pair and connect.

When an LE peripheral supporting this feature is supplied power for the first time, it sends out proprietary Microsoft-defined data in an undirected connectable advertisement. That advertisement is then picked up by the host PC. If the device is in range and its advertisement matches a pattern that is pre-provisioned in the host PC during manufacturing, then the device is paired. This is through Out-Of-Band Pairing which uses a separate secret OOB key that was also pre-provisioned. The distance from the host PC in which the peripheral may pair is determined by another pre-provisioned minimum RSSI value, which is represented in dB, so ranges may vary. All data pre-provisioned must be stored in UEFI to preserve this functionality on clean installs and system restores.

When a bundled device is unpaired within the range defined by the RSSI value, the Bluetooth device automatically re-pairs to Windows. Users must leave the range of the host PC, to be able to pair their device to other Bluetooth enabled PCs.

OEMs must determine on their own how to manufacture these devices. They must also guarantee security and that they will all be in the same box when shipped to customers.

Feature requirements

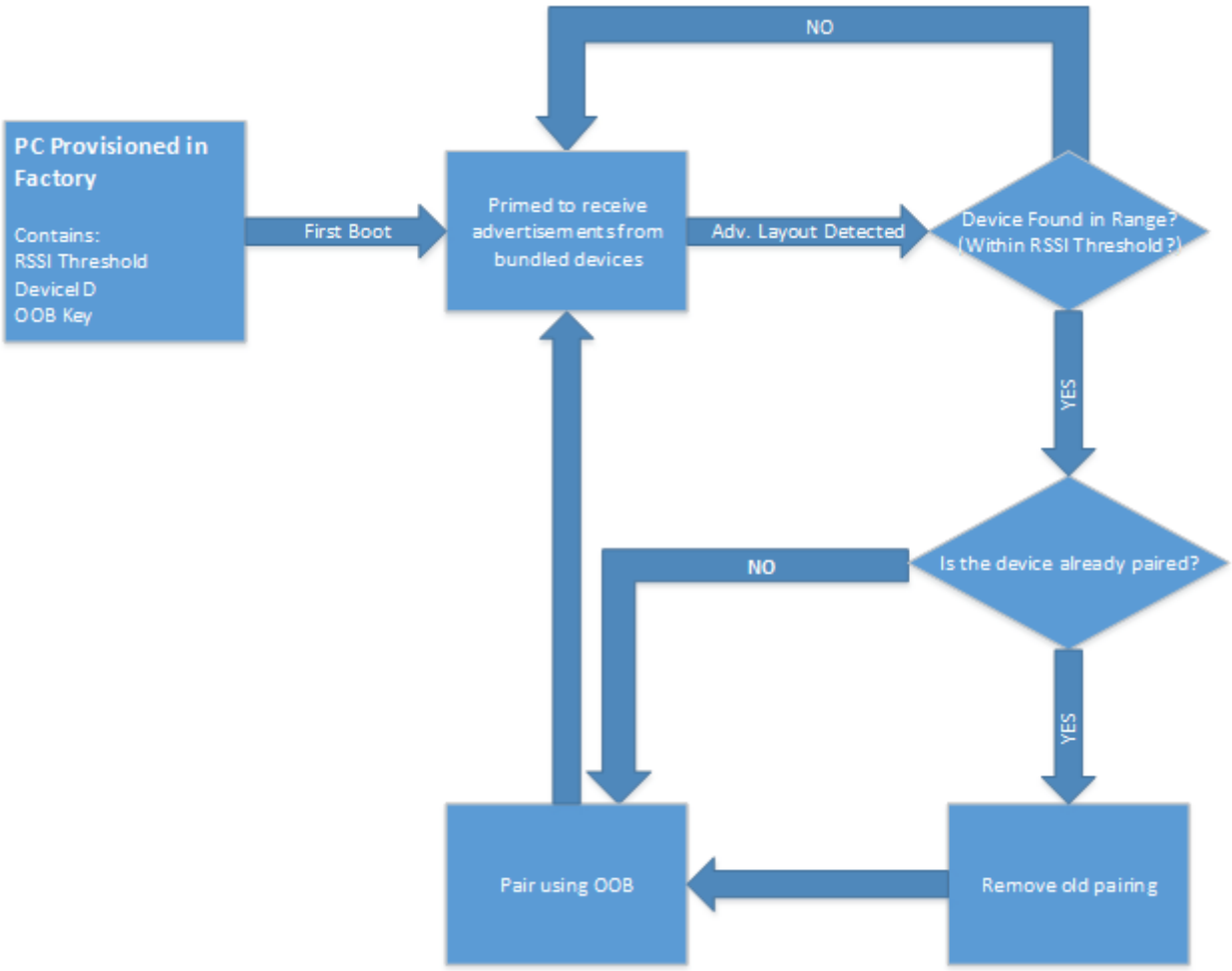
PC

- Windows 10 Anniversary Update
- UEFI Stored Variables
- Microsoft-defined Bluetooth HCI commands for optimized battery life.

Peripheral

- Bluetooth LE

- Store for Device ID (hashed from Bluetooth Address) & TK value
- Custom Advertisement (defined below)



UEFI on host PC

This feature is dependent on storing information to aid the pairing process in UEFI. The feature allows this functionality to persist on system recovery and clean installs. After the UEFI variables are defined, they must be locked to be read-only.

For each device supporting automatic pairing, the host PC should program the following OOB information into its NVRAM:

- RSSI Threshold: Defines a distance threshold in dB for when peripherals are “out of range of the host PC”
  - Set as a minimum signal strength before pairing takes place
  - The OEM must define the appropriate threshold
- DeviceID: uniquely identify a supporting device and should be 8 bytes in length to avoid device conflicts.
  - Must be as unique as the Bluetooth Mac Address
  - Not related to Device ID profile. Only used to represent unique peripherals.

- OOB key: The 16 byte SMP TK value is shared between the host PC and the BTH LE device.

> ⓘ Note
>
> If device ID is not unique per device, or at least per bundle, there will be conflicts if multiple devices are within range of each other. Peripherals pair with unanticipated PCs.

OOB info pseudocode representation

C

```
const unsigned long BTH_LE_DEVICE_ID_SIZE = 8;
#pragma pack(push,1)
typedef struct
{
    CHAR  RssiThreshold;
    UCHAR DeviceId[BTHLE_PREPAIRING_DEVICE_ID_SIZE];
    UCHAR SmpTK[16]; // TK Max Size
} BTHLE_PREPAIRING_ENTRY;
#pragma pack(pop)
```

Public UEFI NVRAM interface definition

C

```
static const LPWSTR BTH_LE_PREPAIRING_NVRAM_VAR_NAME =
L"BluetoothPairingInfo";
static const LPWSTR BTH_LE_PREPAIRING_NVRAM_VAR_GUID = L"{3C901928-0243-
4778-8ADC-BC2D3C6E6B0E}";
```

Advertisement Layout of Peripherals

Section Type [1 Byte]	Manufacturer [2 bytes]	MsftSectionType [4 bytes]	Device Identifier [8 bytes]
0xff (vendor specific)	0x0006 (MSFT)	0x00000004 (Prepairing)	*device identifier*

Security Concerns

During Manufacturing

There is the potential for Man in The Middle attacks if the provisioning data is obtained. Keeping the data secure is up to the OEM.

After the customer has purchased the device

It is possible for the values stored in UEFI to be read, and then spoofed on another device. This can grant access to the system by a malicious user. To mitigate this, we have also created the MDM policy Bluetooth/AllowPrepairing. This policy will allow enterprise and pro deployments to disable this feature.

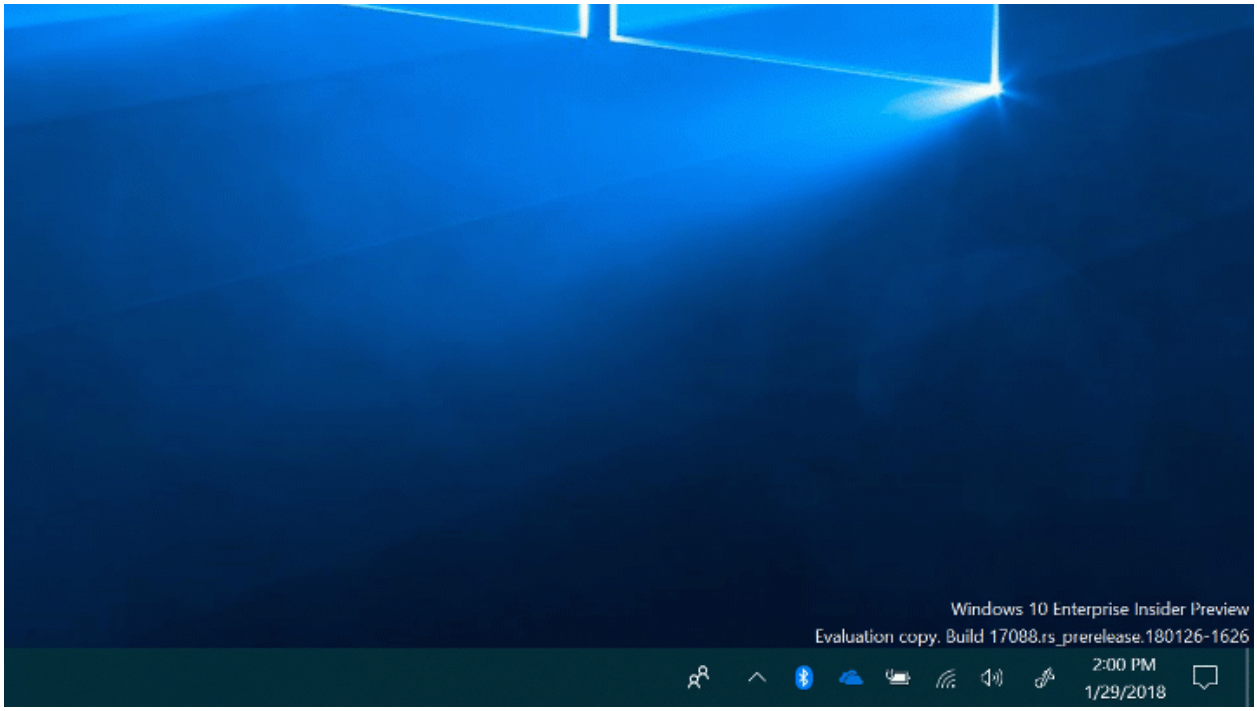
Swift Pair

Article • 06/10/2022

Introduced in Windows 10, version 1803, Swift Pair is the newest way to pair your Bluetooth peripherals to Windows 10 PCs. This topic describes the feature, requirements about how peripherals can support Swift Pair, frequently asked questions about how this quick and convenient feature works.

As the next evolution of pairing, users no longer need to navigate the Settings App and find their peripheral to pair. Windows can now do this for them by popping a notification when a new peripheral is nearby and ready. The steps to use and benefit from this feature are simple:

1. Put the Bluetooth peripheral in pairing mode
2. When the peripheral is close by, Windows will show a notification to the user
3. Selecting "Connect" starts pairing the peripheral
4. When the peripheral is no longer in pairing mode or is no longer nearby, Windows will remove the notification from the Action Center



If at any time, a user wishes to turn Swift Pair on or off, they can do so in the "Bluetooth & other devices" page. Enterprises will also be able to control this feature through the Bluetooth\AllowPromptedProximalConnections in the Policy CSP and with any existing Mobile Device Management solution.

Building a Swift Pair peripheral

There are two sets of requirements to ensure your peripheral works with Swift Pair.

- The peripheral's behavior
- The structure and values in a Microsoft defined vendor advertisement section.

These are the set of requirements:

Peripheral Behavior (Required)

Discovery of Swift Pair peripherals happens over the Bluetooth Low Energy (LE) protocol and **requires the use of LE advertisements**. Windows uses this advertisement to identify a peripheral as Swift Pair capable. This advertisement must contain **one of the Microsoft defined vendor sections** (shown in Fig 2-4) **in the advertisement while in pairing mode**.

In order for Windows to identify a peripheral shortly after it enters pairing mode, **peripherals should beacon for Swift Pair at a faster cadence than normal for quick discovery**. The faster a peripheral advertises; the less time Windows needs to listen. This helps with Bluetooth & Wi-Fi coexistence on the same radio. After a short period of time, the peripheral can fall back to a lower but consistent advertising cadence.

- For quickest discovery: Beacon consistently every 30 ms for >= 30 seconds then fall back to a normal cadence.
- For normal cadence: Beacon consistently every 100 ms or 152.5 ms during a Swift Pair session.
- To keep this experience predictable for our users, the user should not have to try to pair to a peripheral that is no longer available. **Remove the vendor section >=30 seconds before exiting pairing mode**. This is called the "cool-down" phase.

If the peripheral is out of available pairings, **remove the one with the longest time since last connect first**.

Peripheral info on the Swift Pair notification

Users should easily identify the peripheral they are trying to pair to. Peripherals should define **either a defined class of device (CoD) or the peripheral name**, which must be **included in the same advertisement** as the Swift Pair payload. Windows does not active scan due to both power and privacy concerns. As a result Swift Pair **peripheral information cannot be stored in a scan response**.

For LE only peripherals (Fig 2), the Bluetooth SIG **LE appearance section can be parsed to define a class of device**. Windows will parse this section if included in the same

advertisement as Swift Pair and map it to the correct icon to show on the notification. **For dual mode peripherals, the class of device is already covered in the Swift Pair payloads** (Fig 3,4). This is the 3 byte Major/Minor value defined by the Bluetooth SIG.

If a CoD is detected, the displayed icon is the same as the icon shown in Settings.

If a CoD isn't detected, Windows defaults to the Bluetooth logo to show on the notification.

To show a name, **it is recommended to use a Bluetooth friendly name section**, but if that is not possible there is **an *optional* method in the Swift Pair payload** to use the **"Display Name"** field. This field is uncapped in size, but **Microsoft does not localize this string and must work in all markets**. As such, brands, trademarks, or model numbers are recommended to fulfill region market requirements.

If a name is detected, "New [Peripheral Name] found" is shown.

If a name isn't detected, a generic string is shown as defined by the class of device Ex: "New Bluetooth mouse found", "New Bluetooth headphones found", "New Bluetooth headset found"

Spec Features needed for Swift Pair

If a peripheral is beaconing for Swift Pair without any explicit user action, support LE Privacy. Users should not be trackable due to the personal nature of these devices. **If LE Privacy is supported, the peripheral should suspend rotating the Bluetooth LE Address during the Swift Pair session**. The rotated address would be received as a new device request by Windows and would show two notifications for a single peripheral. The Swift Pair session includes the "cool-down" phase: the device address should not be changed until the cool-down phase is complete.

If a dual mode peripheral wishes to pair over both BR/EDR and LE, the peripheral must support secure connections for both protocols. Windows pairs over LE first and derives the BR/EDR keys using secure connections. Pairing to both LE and BR/EDR with Swift pair without the use of secure connections is not supported.

Recommended experiences

For a good first-time experience, enter **pairing mode the first time the peripheral is powered up**. **Do not beacon for Swift Pair indefinitely**. Windows tracks peripherals trying to Swift Pair and will only show one notification per session.

Dual mode peripherals can **save payload space if they pair over both Bluetooth LE and BR/EDR with Secure Connections**.

Payload Structures

Bluetooth LE is required, other BR/EDR peripherals can benefit from Swift Pair. There are three payloads which will trigger Swift Pair; one for solely Bluetooth LE peripherals, and two for dual mode peripherals. The dual mode payloads help address whether the peripheral pairs over both Bluetooth LE and BR/EDR using secure connections or is leveraging this Bluetooth LE advertisement to pair over only BR/EDR.

Octet	0	1	2	3	4	5	6	7		??
Value	0x??	0xff	0x06	0x00	0x03	0x00	0x80	0x??		0x??
Description	Length	Vendor Defined Flag	Microsoft Vendor ID	Microsoft Vendor ID	Microsoft Beacon ID	Microsoft Beacon Sub Scenario	Reserved RSSI Byte	Display Name		

Figure 2: Pairing over Bluetooth LE only

Octet	0	1	2	3	4	5	6	7	8	9	10		??
Value	0x??	0xff	0x06	0x00	0x03	0x02	0x80	0x??	0x??	0x??	0x??		0x??
Description	Length	Vendor Defined Flag	Microsoft Vendor ID	Microsoft Vendor ID	Microsoft Beacon ID	Microsoft Beacon Sub Scenario	Reserved RSSI Byte	Classic Device Icon appearance			Display Name		

Figure 3: Paring over Bluetooth LE and BR/EDR with Secure Connections

Octet	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		??
Value	0x??	0xff	0x06	0x00	0x03	0x01	0x80	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??	0x??		0x??
Description	Length	Vendor Defined Flag	Microsoft Vendor ID	Microsoft Vendor ID	Microsoft Beacon ID	Microsoft Beacon Sub Scenario	Reserved RSSI Byte	Bluetooth BR/EDR Address						Classic Device Icon appearance			Display Name		

Figure 4: Pairing over BR/EDR only, using Bluetooth LE for discovery

Section Header

- This is a Bluetooth SIG defined vendor section
- The header consists of the Length, Vendor-defined flag, and Microsoft Vendor ID 0x0006

Payload Content

Microsoft Beacon ID & Sub Scenario

- The Microsoft Beacon ID helps identify that the advertisement is for this experience and will detail how the peripheral will pair, each unique ID has a unique payload.
- The Microsoft Sub Scenario is a single whose value depends on the type of payload

Value	Payload type
0x00	Pairing over Bluetooth LE only
0x02	Paring over Bluetooth LE and BR/EDR with Secure Connections
0x01	Pairing over BR/EDR only, using Bluetooth LE for discovery

Reserved RSSI Byte

- Reserve this byte and set it to 0x80. This will help maintain forwards and backwards compatibility.

Display Name

- This variable size field can allow payload constrained peripherals to display the name of the peripheral on the notification. This name will not be localized and will need to work for all markets.
- This is field is NOT required and is only a fallback mechanism if a peripheral maker would like to leverage it.

Classic BR/EDR Pairings

For BR/EDR peripheral icon appearance

- It is the same as the Major and Minor Class of Device(CoD) mapping provided by the Bluetooth SIG

BR/EDR Address

- If a peripheral will pair over BR/EDR only, the BR/EDR address in **little endian** format must be included in the main advertising packet
- Supporting Secure Connections and pairing over both Bluetooth LE and BR/EDR removes this requirement

Turning Swift Pair on by default

Swift pair released starting with Windows 10 version 1803 but was not turned on automatically for users. This decision was made when we learned continuously monitoring Bluetooth Low Energy (LE) advertisements caused some radios to improperly handle Wi-Fi activity when on the same radio. Swift Pair is the first of many planned Windows features where the system will be continuously monitoring Bluetooth advertisements. To address this, we now have a way for radios to report that they can handle these concurrent scenarios.

Hardware diversity in the ecosystem prevents us from testing against every PC, so Windows needs to rely on each radio to declare support for continuous monitoring. Declaring support implies that the radio has been tested to minimize drops in Wi-Fi's performance when continuously monitoring advertisements. Radios that can adequately perform Wi-Fi activities while monitoring Bluetooth LE advertisements concurrently can declare support by setting the appropriate value in HCI_VS_MSFT_Read_Supported_Features commands.

General monitoring of Bluetooth LE advertisements (not continuous) is a prerequisite to supporting continuous monitoring. It is expected that most Windows certified radios already support the vendor specific command for general monitoring. General monitoring (0x8) enables Swift Pair in Settings but does not automatically turn on Swift Pair. A user must find the setting and turn it on themselves.

If the radio supports continuous monitoring of LE advertisements, the radio can declare support via the 0x20 bit in the HCI supported features command. Once this bit is set, Swift Pair will be turned on automatically, and a user no longer needs to go to the Settings page to do so.

If you are trying to enable Swift Pair automatically on any other continuous monitoring feature, see the testing considerations for concurrent Wi-Fi and Bluetooth scenarios below.

Testing Considerations

- Wi-Fi throughput, jitter, and latency

    - Measure throughput, jitter, and latency in strong, medium, and weak RSSI conditions.
    - Measure with multiple client devices connected to the Access Point to simulate typical real-life usage.
    - We also recommend running your Wi-Fi and Miracast Interop and Connectivity tests with the Swift Pair feature turned ON and OFF.
- Battery life
    - Performance measures for battery life should be evaluated through the hardware vendor's proprietary power and battery tests.
- Wi-Fi and Miracast connection reliability
    - Performance measures can be evaluated after Windows driver flighting.
- Wi-Fi access point roaming time
    - Performance measures can be evaluated after Windows driver flighting.
- Miracast stream Quality
    - Performance measures can be evaluated after Windows driver flighting.

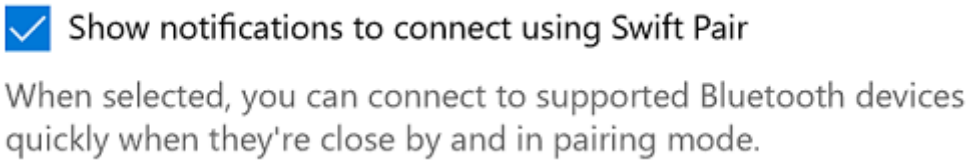
Frequently Asked Questions

If I make a Bluetooth peripheral, do I need to get it certified by Microsoft to use Swift Pair?

You are free to use Swift Pair. No certification is needed for this experience.

I put my Swift Pair-enabled peripheral in pairing mode, and nothing happens. What do I need to do?

In Windows, version 1803, you must enable Swift Pair. On **Settings**, search for **Bluetooth & other devices**. Check the **Show notifications to connect using Swift Pair** box.


Show notifications to connect using Swift Pair
When selected, you can connect to supported Bluetooth devices quickly when they're close by and in pairing mode.

Does this mean the similar experiences on other platforms will work on Windows?

At this time, only certain enabled peripherals can trigger Swift Pair. Check back for updates.

I can't get a notification to show, and I don't have an option in Settings. What is going on?

If the **Show notifications to connect using Swift Pair** option is not shown, then the Bluetooth radio in your Windows device does not have the required hardware support. Please let us know through the Feedback Hub.

How does Windows detect if a peripheral is within range?

Windows, version 1803 also introduces a Bluetooth proximity service that reads the signal strength from a peripheral to determine its proximity to Windows. When the peripheral is considered within range, the notification is shown in **Settings**.

Because the signal strength varies between hosts and peripherals, we are always looking to increase the precision of the service to make the proximity detection more accurate. This feature is based on the the reserved Received Signal Strength (RSSI) byte in the payload.

How does Windows listen for these peripherals without draining power?

The pattern that Windows looks for is offloaded to the radio through Hardware Offload. The offloaded pattern listens for a match to the vendor section without waking up the system or active scanning. If the radio does not support Hardware Offload, then the feature is not supported, and this is represented by not showing **Show notifications to connect using Swift Pair** in **Settings**.

How does Windows detect and display Bluetooth peripheral information?

The name and the type of peripheral must be in the same advertisement that has the vendor section. Windows does not active scan for this feature, and all peripheral information must be included in this single advertisement. If the friendly name section can't fit in this advertisement, a fallback mechanism is provided at the end of the payload to show the peripheral name.

How can an OEM enable Swift Pair by default when the radio does not declare support?

An OEM can enable Swift Pair by default by using the registry subkey below if the recommended validation in Testing Considerations has been completed. This may be required if the radio manufacturer has not declared support for HCI_VS_MSFT_Read_Supported_Features commands in the radio firmware.

Console

```
    Subkey:
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\SwiftPair
    Value: SwiftPairDefault
    Type: REG_DWORD
    Data: 1
```

Display

Article • 12/17/2021

Scaling

- Unified scaling system. Scaling has been unified between most frameworks and between SKUs. Windows 10 scale factors are: 100%, 125%, 150%, 200%, 250%, 300%, 400%, and 500%. The following will all use these scale factors:
  - Windows 10 Universal apps (UAPs)
  - DPI-aware Win32 apps (asset quality dependent on app)
  - System UI on Mobile and Desktop SKU
  - Edge and Internet Explorer content (user can override in-app)
- Expanded scaling system for 4K and 8K devices (up to 500%)
- Universal Windows Platform (UWP) apps with their good per-monitor scaling are first-class desktop apps
- Lower minimum resolution for very low end devices running desktop SKU
- More desktop UI scales on per-monitor basis

Refresh rate

Dynamic refresh rate switching for media content

On supported hardware, Windows 10 automatically and seamlessly switches the refresh rate to better match the frame rate of any full-screen videos that are playing. For example, when playing a 24fps video (film), Windows 10 switches to 24 Hz or 48 Hz. When playing a 30fps video (NTSC TV), Windows 10 switches to 30 Hz. This functionality provides two key benefits to the user:

1. For film content (24fps), by setting the refresh rate to an integer multiple of the video frame rate, 3:2 pulldown no longer needs to be performed. Each video frame will be displayed for exactly 41.7ms – 3:2 pulldown, which is necessary at 60 Hz (since it is 2.5x the frame rate), results in a pattern of 33.3ms, 50ms, 33.3ms, 50ms, …, causing ‘film judder’.
2. Because the optimal refresh rate can be lower than the standard 60 Hz, running at these refresh rates actually improves video playback battery life for users. Running

at 48 Hz for 24 fps video improves system battery life upwards of 5%, while running at 24 Hz provides an additional 10% savings over 48 Hz.

Because the battery life savings when running at 24 Hz are so significant (measured in hours), we recommend that all portable form factors use displays that are qualified for 24 Hz and 30 Hz refresh rates when possible. Generally, this means using display panels built on the Indium-Gallium-Zinc-Oxide (IGZO) process. Amorphous Silicon-based panels (a-Si) are generally only able to support the 48 Hz and 50 Hz refresh rates (which are still valuable).

Windows 10 switches to the following refresh rates based on the underlying content in order of priority (lower refresh rates are always a higher priority since the power savings are greater there).

24 fps (NTSC film)	24 Hz, 48 Hz, 60 Hz
25 fps (PAL film)	25 Hz, 50 Hz, 60 Hz
29.97 fps (NTSC TV)	30 Hz
48 fps (very rare, some films shot at 48 fps)	48 Hz
50 fps (PAL TV)	50 Hz
59.94 fps (NTSC TV)	60 Hz

This feature is designed so that the lower refresh rates are triggered automatically and triggered without any visible artifacts normally associated with a mode switch.

For this feature to be engaged by Windows 10 at a particular refresh rate, the system must report support for the rate through its firmware. By reporting support for a particular refresh rate, the system is guaranteeing that the display panel can run at the corresponding refresh rate without any adverse effects (such as flicker).

Adaptive sync for gaming

On supported hardware, Windows 10 (version 1903 and later) supports automatically synchronizing the refresh rate to the presentation rate of games running fullscreen.

High dynamic range (HDR) and wide color gamut (WCG)

On supported hardware, Windows 10 supports high dynamic range (HDR) displays which may optionally have a color gamut wider than sRGB. This is referred to as

Windows Advanced Color. When Advanced Color is active on a display, Windows 10 will automatically manage the colors for all applications in order to ensure color accuracy and consistency.

Windows requires that HDR displays conform to a set of minimum guidelines:

Applicability	Requirement
For all displays	VESA DisplayHDR 1.1 certification required
For LCD displays	VESA DisplayHDR 600 or better recommended
For OLED or micro-LED displays	VESA DisplayHDR 500 True Black or better recommended

For LCD displays with multiple backlight dimming zones (highly recommended for LCD displays), Microsoft recommends using a 2-dimensional array of zones rather than only horizontal or vertical zones.

Display descriptors

The Windows display stack relies on accurate information reported in a display's hardware descriptor. Windows supports VESA Extended Display Identifier (EDID) descriptors, as well as limited support for DisplayID descriptors.

For all descriptors, Windows depends on the following pieces of information (read directly from the descriptor):

Descriptor feature	Requirement	Notes
Physical size	Required	Used for DPI scaling calculations. Projectors may set the physical size to zero.
Native resolution	Required	Used for DPI scaling calculations and selecting the default mode
Unique serial number per device	Required	Used to track user preferences for individual displays
Unique model name	Required	Used to identify models for providing friendly names, monitor drivers, and OEM overrides
Colorimetry data (e.g. white point and color primaries)	Required for HDR and WCG displays	Tolerance is expected to have $\Delta E < 5$
Luminance and HDR static tone mapping data	Required for HDR and WCG displays	

Windows specifically supports the following display descriptor formats for the above features:

Descriptor format	Windows 10 support	Windows 11 support	Requirement
EDID 1.4 base block	Supported	Supported	Required for displays with an EDID
CEA-861 extension blocks	Supported	Supported	
CTA-861.3-A HDR static metadata extension	Supported	Supported	Required for HDR displays with an EDID on Windows 10, see below for HDR descriptor requirements
DisplayID 1.3 (as an EDID extension block)	Not supported, ignored	Not supported, ignored	Graphics drivers may support extended timing information
DisplayID 2.0 (in standalone block)	Supported	Supported	See below for DisplayID 2.0 requirements
DisplayID 2.0 (as an EDID extension block)	Not supported, ignored	Supported	See below for DisplayID 2.0 requirements

HDR display descriptor requirements

Windows 10 does not support DisplayID 2.0 as an EDID extension block, so HDR displays should use an EDID with a CTA-861.3-A HDR static metadata extension, *or* a standalone DisplayID 2.0 block without an EDID.

Windows 11 adds support for DisplayID 2.0 as an EDID extension block, but requires that HDR properties be specified using a DisplayID 2.0 Display Parameters block for colorimetry and a DisplayID 2.0 Display Features block for EOTF support. Windows 11 does not support HDR parameters to be specified in a CTA-861.3-A embedded in a DisplayID sub-block.

DisplayID 2.0 requirements

Windows does not support DisplayID 1.x blocks, and always ignores them.

Windows 10 has limited support for standalone DisplayID 2.0 descriptors. These descriptors are only utilized if the monitor has no EDID.

Windows 11 adds support for qualifying DisplayID 2.0 descriptors embedded in EDID extension blocks. Windows 11 also now prefers qualifying DisplayID 2.0 descriptors over EDID for all properties other than the PnP identification. In other words, in the presence of a qualifying DisplayID 2.0 descriptor, legacy EDID blocks are assumed to be deprecated legacy blocks maintained only for older systems. Therefore, it's important that a DisplayID 2.0 descriptor *fully* encompass all properties to identify and describe a monitor.

The DisplayID 2.1 specification adds clarifications around requirements for DisplayID descriptors, both embedded in an EDID extension and standalone. Windows 11 requires that DisplayID 2.0 descriptors follow these requirements precisely in order for the descriptor to be considered a "qualifying" DisplayID 2.0 descriptor. If Windows finds that the DisplayID block does not meet these requirements, the block is entirely ignored and the system falls back to only using the EDID. In particular:

- The DisplayID container must have version 2.0 or later.
- The first sub-block must be a valid Product Identification sub-block.
- There must be at least one Type VII timing sub-block as the first mode block declaring the native mode.
- There must be a valid Display Parameters sub-block.

Windows 11 prefers descriptors in the following order:

1. Standalone DisplayID 2.0 descriptor (if qualifying)
2. DisplayID 2.0 descriptor embedded in EDID extension blocks (if qualifying)
3. EDID

> ⓘ **Note**
>
> Since qualifying DisplayID descriptors are required to have all information necessary to replace the EDID, DisplayID descriptors may need to span multiple EDID 128-byte extension blocks. Manufacturers are recommended to pick panels with support for at least 384 bytes (3 blocks) of storage to accommodate descriptor changes.

> ⓘ **Note**
>
> Graphics drivers may make use of DisplayID features or descriptors that Windows does not currently support as long as Windows provides a higher-level DDI that allows the driver to express the functionality without a Windows DisplayID parser dependency.

Connectivity standards

DisplayPort and Embedded DisplayPort (eDP)

On supported hardware, Windows 10 and Windows 11 support displays connected via the DisplayPort standard, including daisy-chaining and multi-stream topology (MST) configurations with multiple monitors attached to a hub device.

> **Note**
>
> Windows 10 and Windows 11 do not provide native support for multi-SST operation (MSO) over eDP. Since MSO displays do not provide a single display descriptor for the entire display, use of MSO eDP displays is known to result in undefined behavior and is not recommended.

Miracast

On supported hardware, Windows 10 and Windows 11 support wireless projection over Miracast. Find out more.

Related topics

GetSystemMetrics

Windows Hardware Compatibility Program

Unattend settings (Windows 8.1)

Touch displays

Article • 11/05/2020

Windows 10 Guidelines

This topic covers guidelines for Touch display components in Windows 10.

Touch requirements

The following table lists Windows 10 features related to touch.

Desktop and mobile use same input platform	• No need to maintain separate drivers per form factor • Firmware solutions always work, always up to date
Touch experience significantly improved	• Desktop is more touch-friendly than ever • Touch targeting has now been brought to mobile
Lower requirements that complement Windows 10 touch experiences	• Always procure high quality components to ensure best user experience

The following table lists requirements for touch-related attributes:

Attribute	**Windows 10 Requirements**
HID Compliance	Required
Ghost Points	No ghost contacts are reported
Accuracy	<= +/- 1mm (outside 3.5mm from edges) <= +/- 2mm (within 3.5mm of all edges)
Move Latency	For <7”, <= 35ms For >=7”, <=25ms
Down Latency (Active)	<= 35ms
Down Latency (Idle)	<= 150ms
Cold Boot Latency	Immediately responsive once the display is active

Attribute	Windows 10 Requirements
Resolution	>= Native display resolution or greater
Linearity (Moving jitter)	<= +/- 1mm (outside 3.5mm from edges) <= +/- 2mm (within 3.5mm of all edges)
Stationary Jitter	<= 0.5mm
Finger Separation	<= 8mm [Edge to Edge]
Minimum Simultaneous Contacts	>= 5
Buffering	>= 100ms of data
Report Rate	>=Display Refresh Rate <=250Hz
Custom Gestures	Custom gestures designed to work with user interface elements are prohibited
Third party drivers	Desktop: Third party drivers prohibited Mobile: Third party drivers allowed

Touch implementation

Drivers

- Drivers built for Windows 8.1 phone and desktop continue to work on the updated touch platform
- Driverless, firmware-based solutions ensure that touchscreens always work
- These solutions will be seamlessly upgraded to Windows 10

Compatibility

- Desktop touch devices previously certified for Windows 8 and Windows 8.1 are automatically fully compatible
- Mobile devices that met the chassis spec guidelines need to be re-tested
- Touch solutions are backed with a hardware compatibility program that ensures a great experience

TEST JIGS FOR TOUCH HLK

Most touch tests are identical to the existing test for Windows 8.1, with upgraded parameters:

- Existing testing tools from Windows 8 and Windows 8.1 are still used for Touch tests:
  - Precision Touch Testing Tool (PT3)
  - Rotational Acoustic Tool (RA)
- Windows 10 accessory jigs to enable Touch testing:
  - 7mm slugs
  - 16mm slug

Maintained guidelines from Windows 8.1

Display surface

The following guidelines are provided to create a successful touch experience on touch-enabled displays.

- Use glass or glass coatings designed to reduce fingerprints.
- Consider anti-glare materials and LED-based illumination to ensure screen readability in outdoor and brightly lit indoor environments.
- Choose an anti-glare material with the following characteristics:
  - Low haze value (<=6%) to minimize reduction of display clarity and contrast while providing minimal friction.
  - Finer than the sub-pixel pitch to prevent sparkling.
- Minimal surface friction (surface roughness of 100-500 nm (RMS). High surface friction causes the finger to skip over the surface, breaking touch contact.

Device cover glass

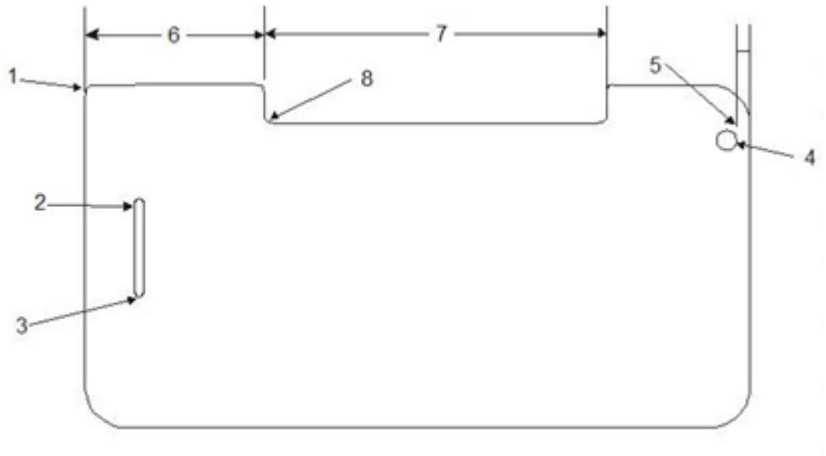
This section defines the functional attributes for device cover glass that provides the user with a high quality touch-screen experience worthy of the Microsoft brand. Attributes include those that preserve and protect the surface, appearance, and device, and improve the touch functional experience.

Cover glass for discrete applications

Discrete cover glass applications use the glass as a protective display cover on top of the touch sensing layer, but not as a physical carrier or substrate for the touch sensing layer (ITO, etc.) itself. You should conduct all tests and measurements following the conditions outlined in Cover Glass Test and Measurements.

For optimal damage resistance:

- The cover glass should be chemically-strengthened with a minimum magnitude and depth of layer (DOL) of the compressive stress as follows. In all cases, glasses should exhibit non-frangible behavior. Frangible behavior is when the glass breaks into a large number of small pieces when hit with sufficient force:
  - 0.55 mm: ≥ 700 MPa CS and > 35 microns DOL
  - 0.7 mm: ≥ 750 MPa CS and > 40 microns DOL
  - 1.0 mm: ≥ 750 MPa CS and > 45 microns DOL
- 4-point bend test performance (edge strength) recommendations:
  - 0.55 mm: Average peak stress = 600 MPa
  - 0.7 mm: Average peak stress = 620 MPa
  - 1.0 mm: Average peak stress = 620 MPa
- Abraded ring-on-ring test performance (surface strength) recommendations:
  - 0.55 mm: Average load-to-failure = 30 kgf
  - 0.7 mm: Average load-to-failure = 55 kgf
  - 1.0 mm: Average load-to-failure = 100 kgf
- The following general design guidance should apply to machined cover glass parts:



#	Feature	Measure	Guidance
1	Outer corner	Radius	> 1.0 mm
2	Slot radius	Radius	> 1.5 mm

#	Feature	Measure	Guidance
3	Slot width	Width	> 1.5 mm
4	Min. hole diameter	Diameter	> 1.5 mm
5	Hole-to-edge distance	Diameter	> 4.0 mm
6	Width of protrusion	Width	Width > Depth
7	Width of the slot	Width	Width > Depth
8	Inner radius	Radius	> 1.0 mm

Note Tests conducted using a standard 1mm glass.

- The indentation threshold as measured with a Vickers indenter should be ≥ 5* kgf.
- The Knoop scratch load to lateral cracks should be ≥ 4* N.

High-Quality Touch Experience Recommendations:

- Minimum dielectric constant is 7.0.
- The Young's Modulus of the glass should be 71.5±5 GPa.
- The coefficient of thermal expansion (CTE) should be 80±4 x 10^{-7} per degree C.
- Water absorption should meet Hydrolytic Resistance Class 2 or better.

High-Quality Viewing Experience Recommendations:

- Optical transmission as measured through 1.0 mm thick cover glass is >91% nominal transmission across the 390-760 nm (visible) wavelength spectrum, with variation not to exceed ±1%.

Cover glass for integrated touch applications

Integrated touch cover glass applications are glass covers that serve as a protective display cover and as a physical carrier or substrate for the touch sensing layer (ITO, etc.) itself. Integrated touch cover glass applications are also known as one-glass solutions (OGS). You should conduct all tests and measurements following the conditions outlined in Cover Glass Test and Measurements.

Optimal Damage Resistance Guidelines:

- The cover glass should be chemically-strengthened with a minimum magnitude and depth of layer (DOL) of the compressive stress (CS) as follows. In all cases,

glasses should exhibit non-frangible behavior. Frangible behavior is when the glass breaks into a large number of small pieces when hit with sufficient force:
    - 0.55 mm: ≥ 500 MPa CS and > 25 microns DOL
    - 0.7 mm: ≥ 550 MPa CS and > 37microns DOL
    - 1.0 mm: ≥ 550 MPa CS and > 55 microns DOL

- 4-point bend test performance (edge strength) recommendations:
    - 0.55 mm: Average peak stress = 600 MPa, B10 > 450MPa
    - 0.7 mm: Average peak stress = 620 MPa, B10 > 500MPa
    - 1.0 mm: Average peak stress = 620 MPa B10 > 500MPa

- Abraded ring-on-ring test performance (surface strength) recommendations:
    - 0.55 mm: Average load-to-failure = 13 kgf
    - 0.7 mm: Average load-to-failure = 20 kgf
    - 1.0 mm: Average load-to-failure = 29 kgf

- Refer to the figure for the machined cover glass part for general design guidance on machined cover glass parts. Holes and/or slots are not recommended on cover glass used for integrated touch applications due to compromises in edge strength.

- The indentation threshold as measured with a Vickers indenter should be ≥ 5 kgf.

- The Knoop scratch load to lateral cracks should be ≥ 4 N.

High-Quality Touch Experience Guidelines:

- Minimum dielectric constant is 7.0.
- The Young's Modulus of the glass should be 71.5±5 GPa.
- The coefficient of thermal expansion (CTE) should be 80±4 x 10^{-7} per ºC.
- Water absorption should meet Hydrolytic Resistance Class 2 or better.

High-Quality Viewing Experience Guidelines:

- The recommended optical transmission as measured through 1.0 mm thick cover glass is >91% nominal transmission across the 390-760 nm (visible) wavelength spectrum, with variation not to exceed ±1%.

Cover glass tests and measurements

- Make all measurements on bare glass with no coatings, films, or other types of surface treatments applied.

- Conduct all tests in a controlled environment (23±2º C, 50±5% RH).

- 4-Point Bend

Perform horizontal bending testing using 18mm loading spans, and 36mm support spans applying a nominal crosshead rate of 5mm/min. The preferred sample geometry is 44mmx60mm. Breaking stress is reported based on ASTM C158. Sample geometries beyond the preferred geometry may require consultation by Corning on span selection.

- Abraded Ring-on-Ring (AROR)

  Abrasion with 90 grit Silicon Carbide @ 5psi, 5 seconds, ¼" mask; retained strength measured through Ring on Ring, ½" load ring, 1" support ring. Nominal crosshead rate of 1.2mm/min. Center the abrasion on the glass sample and place it in the center of the loading ring for testing. Breaking load is reported. The preferred sample geometry is 50mmx50mm. You can use ASTM C1499 as a reference for some aspects of the ring on ring procedure.

- Indention

  A Vickers indenter makes a series of indents in a glass samples, stepping through a range of repeated loads and held at the maximum load for 10 seconds, samples are inspected to assess the load where >50% of the indents exhibit evidence of radial cracks after a fixed period of time once the indents have been created. Loading/unloading rates = 0.2mm/min.

- Scratch Threshold

  A Knoop indenter places a series of 10mm scratches in a sample. Repeated scratches are performed over a range of loads, samples are inspected to assess the load where >50% of the scratches exhibit evidence of lateral cracks after a fixed period once the scratches have been created.

Related topics

Human Input Devices Design Guide

Windows Hardware Compatibility Program

HID over I2C

Unattend settings (Windows 8.1)

Guidance for Rounded Display Bezels

Article • 12/02/2021

Overview

As part of Windows 11's unique new design language, rounded corners play an integral role in the look and feel of the UI. The overall perception of rounding in the system is shaped by both software rounding, controlled by the OS, and hardware rounding, controlled by OEM partners. This topic provides Microsoft guidance for OEM partners regarding the rounding of hardware display bezels to achieve the best user and developer experience.

While OEM partners are free to differentiate their hardware designs with varying hardware rounding radii and display properties, the operating system only supports the rounding that ships with Windows 11. We do not support arbitrary radii for software to match arbitrary radii for hardware due to high complexity in system implementation, and to avoid placing the burden on the app developer ecosystem to program against an indefinite number of software and hardware radii combinations. Therefore, this topic will help you calculate the effective pixels that your hardware rounding will clip based on the display scale factor so you can know what range of hardware radii to safely use without degrading the UI.

> **Note**
>
> All scale factors in this topic refer to Desktop scaling.

Software Rounding Linear Scaling

Windows 11 ships with two radii available for rounding – 4 px for controls such as buttons and input fields, and 8 px for flyout menus and main app windows. For the purposes of this topic, the 8 px radius is the default software rounding radius.

The design of software rounding in Windows 11 is based on an 8 px radius at 96 DPI, and this scales linearly as shown in the following table:

DPI Scale Factor	Window Pixels Rounded
100%	8
150%	12

DPI Scale Factor	Window Pixels Rounded
200%	16

This simple linear relationship serves to illustrate how Windows scales its rounding, but ultimately how much clipping occurs depends on the native resolution of the display and what the scale factor is. For example, a display with a default scale factor of 200% will clip the UI less than the same display set to 100% because 100% would make the content smaller compared to the intended 200%. On this example display, one effective pixel is the same as one physical pixel at 100%, thus more pixels are being clipped.

Calculating the Windows Default Scale Factor and Effective Resolution

The amount of UI clipping that rounded bezels incur is affected by the scale factor of the OS and the display's resolution and size, a default for which is chosen by the system based on the following formulas and variables.

> **ⓘ Important**
>
> Because almost all users do not change the default scale factor, it should be the baseline for all calculations relating to UI clipping by rounded bezels.

Pixel Density

The first variable in the default scale factor formula that Windows needs to know is the density of pixels per inch (DPI) of the display. This is given by the following formula and is predetermined by the display manufacturer:

$$Native\ DPI = \frac{\sqrt{(Resolution\ x)^2 + (Resolution\ y)^2}}{Size\ (in\ inches)}$$

Viewing Distance

Because viewing distance affects how large objects on screen appear to the user when combined with native DPI, the system factors it into its calculations when deciding on a default scale factor. The viewing distance of a display is roughly based on its physical size and device type, as shown in the following table:

> ⓘ **Note**
>
> This list is not exhaustive for all possible device types.

Device Type	Viewing Distance (in inches)
Small Tablet	16.3
Tablet	20
Laptop	24.5
Desktop	28
TV	84

Optimal Zoom

Optimal zoom is the precise decimal level of scaling that Windows believes is best for a display. This maps roughly to the default scale factor, which takes the optimal zoom and rounds to the nearest 25% increment within a scoped range. Windows computes the optimal zoom level for a display based on the native DPI and viewing distance. Note that 96 is the default DPI and 28 is the default viewing distance in the system.

$$Optimal\ Zoom = \frac{(Native\ DPI * Viewing\ Distance)}{(96 * 28)}$$

Max Scale Factor

The final information Windows requires to determine a default scale factor is the max scale factor of a display, or the level above which it will not scale content to prevent accessibility problems (for example, a message box so large that the OK button is off the screen). The max scale factor is determined by the vertical resolution of the display.

Vertical Resolution	Max Scale Factor
< 900	100%
>= 900 and < 1080	125%
>= 1080 and < 1440	150%
>= 1440 and < 1800	200%

Vertical Resolution	Max Scale Factor
>= 1800	250%

Default Scale Factor

Finally, the default scale factor is decided based on the optimal zoom. Effectively, the default scale factor simply rounds the optimal zoom to the nearest 25% increment, with a minimum of 100% and a maximum of the Max Scale Factor.

Optimal Zoom	Default Scale Factor
< 1.2	100%
>= 1.2 and < 1.43	125%
>= 1.43 and < 1.78	150%
>= 1.78 and < 2.32	200%
>= 2.32 and < Max Scale Factor	250%
> Max Scale Factor	Max Scale Factor

Calculating Effective Resolution

Once the default scale factor is known, the native resolution is scaled to determine the effective resolution for the display.

$$Effective\ Resolution = \frac{Native\ Resolution}{Scale\ Factor}$$

Walkthrough - Measuring Effective Pixels Clipped by Rounded Hardware Bezels

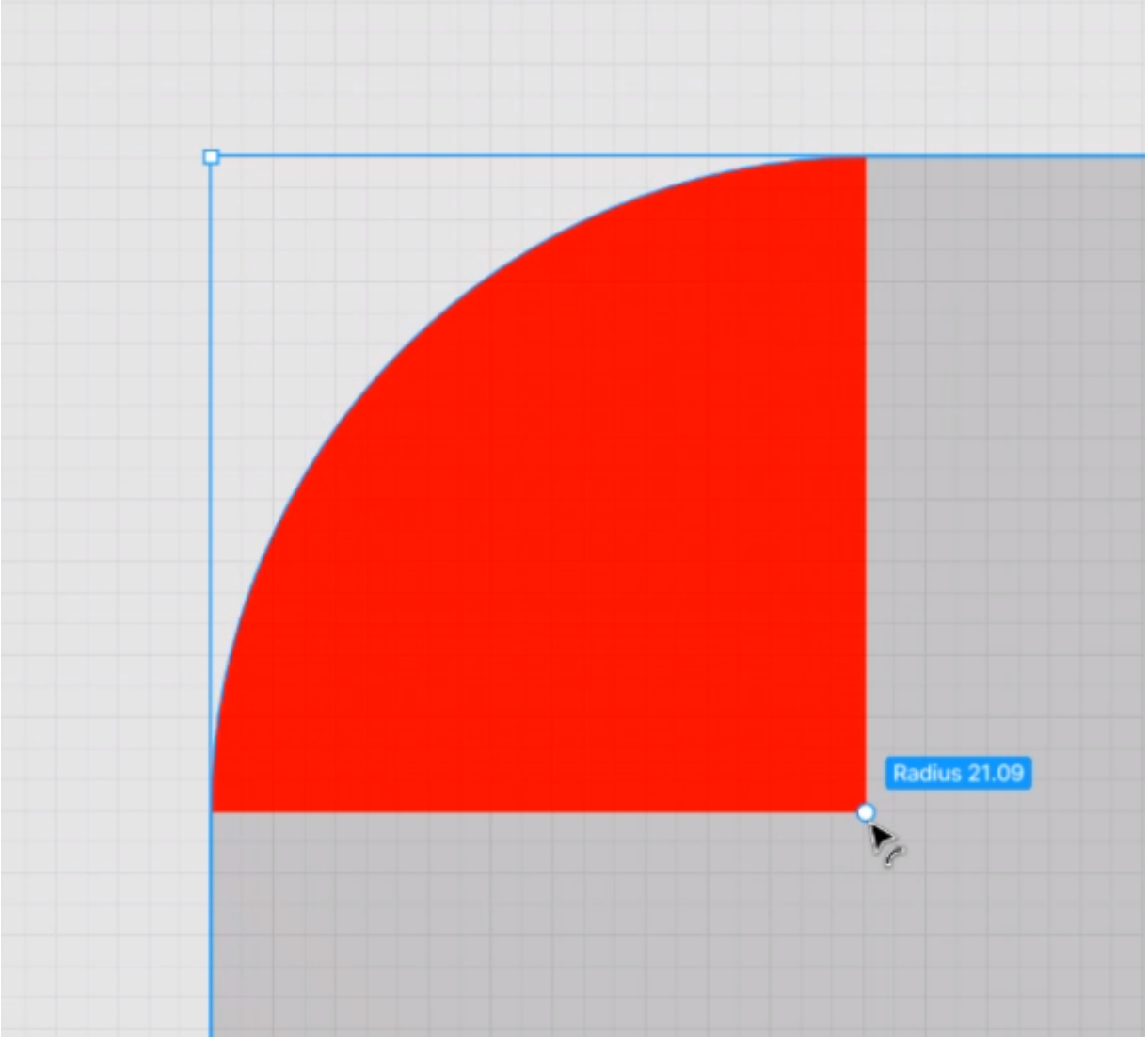
When evaluating the UI impact of different curvatures and radii of rounded display corners, the first result to measure is the physical pixels that are clipped by the bezel at the native resolution, which can then be divided by the scale factor. In this section, we will walk through an example of measuring the physical pixels clipped for a hypothetical display with the following properties:

Property or Calculation	Value

Property or Calculation	Value
Screen Size	12.4 inches
Horizontal Resolution	2560
Vertical Resolution	1600
Native DPI	243.46
Viewing Distance	24.5 inches (Laptop device type)
Optimal Zoom	2.2
Max Scale Factor	200%
Default Scale Factor	200%
Target rounding radius	2.2 mm

Circle Formed by the Rounded Bezel

For this example, we would like to round the bezel by 2.2 mm. The arc produced by this amount of rounding forms a quarter circle that can be overlaid with the corner of the normal display rectangle, highlighted in red in the following image:



The radius for the complete circle is calculated with the following formula:

$$Radius\ of\ Circle\ Formed\ by\ Rounded\ Bezel = (\frac{Native\ DPI}{25.4}) * (Desired\ radius\ in\ mm)$$

25.4 is the conversion of millimeters to inches, because displays and DPI are measured in inches while rounding radii are measured in millimeters. For our example display, this results in a radius of **21.09 px**. The total area of this circle is given by the standard circle area formula:

$$Area\ of\ Circle = \pi r^2$$

Our example display's rounded bezel circle has an area of **1395.84 px**.

Corner Square

The square that surrounds the rounding circle has a side length equal to the circle's diameter, calculated as follows:

$$Side\ of\ Corner\ Square = (Radius\ of\ Circle\ Formed\ by\ Rounded\ Bezel) * 2$$

The area of this square is given by the standard square area formula:

$$Area\ of\ Square = Side^2$$

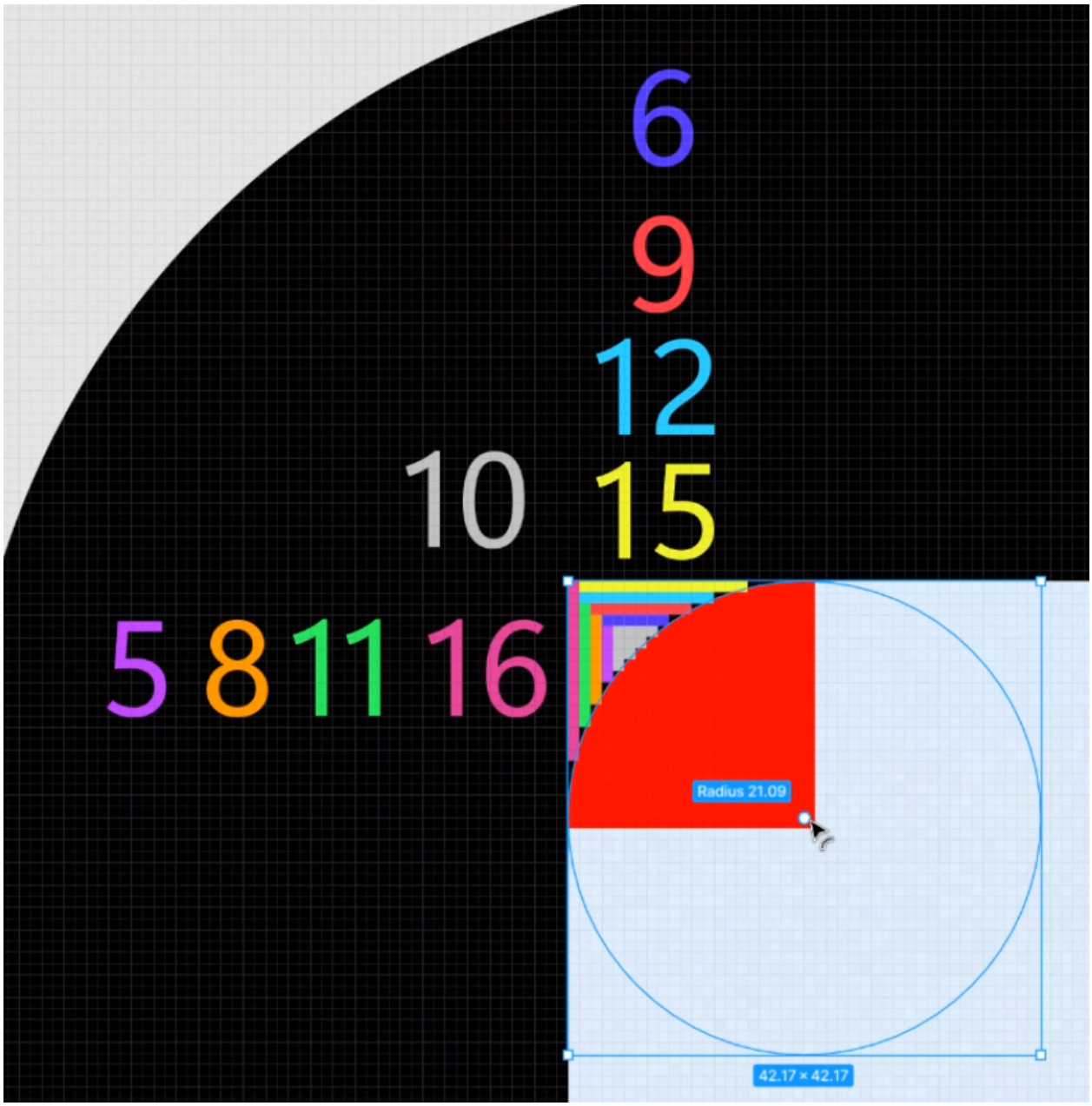
For our example, the square has a side length of **42.17 px** and an area of **1778.51 px**.

Physical Pixels Clipped per Corner

With the radius of the corner circle known, we can calculate the total area clipped by this circle and divide by 4 to obtain the pixels clipped in the display corner.

$$Physical\ Pixels\ Clipped\ per\ Corner = \frac{(Area\ of\ Corner\ Square) - (Area\ of\ Circle\ Formed\ by\ Rounded\ Bezel)}{4}$$

The number of pixels clipped per corner for our example display is **95.42 px**, which includes fractional pixels. In the following image, the desired quadrant area is highlighted by various colors to very roughly illustrate whole pixels clipped by the rounded bezel. This image also shows the outlines of the corner circle and the corner square that surrounds it.



The approximate number of whole pixels clipped in this example is **92 px** for this corner of the display.

Effective Pixels Clipped

Finally, we can divide the physical pixels clipped by the scale factor to determine how many effective pixels are occluded.

$$Effective\ Pixels\ Clipped = \frac{Physical\ Pixels\ Clipped}{Scale\ Factor}$$

With a default scale factor of 200%, this results in **47.72 epx** clipped per corner for our example.

Recommended Rounding Radii

The amount of UI clipped by a display's rounded bezels is affected by the desired rounding radius, which determines the radius of the bezel corner circle.

> ⚠ **Warning**
>
> Based on internal testing on a device with 3 mm rounding radius, **the recommended range of corner circle radius lengths that works with Windows 11's rounding design is 16-24 effective pixels (epx). Any radius above of this range can degrade the UI and is not supported. Any radius below this range will not create UI clipping concerns, but could create concentricity tension between the rounding angles of the display and the UI.** In other words, the difference between the bezel corner radius and the window rounding radius will begin to differ greatly and will begin to impact the user's visual perception of the system.

Taking the 200% default scale factor into account, the example display from our walkthrough has a rounding circle with an effective radius of **10.55** epx. Therefore, the UI will not be clipped excessively, but the difference in rounding angle betewen the bezel and the Windows 11 rounded window corners must be taken into account.

Dynamic Lighting devices

Article • 07/09/2024

> **Note**
>
> For a user experience walkthrough and FAQ for Dynamic Lighting, see: Support | Control your Dynamic Lighting devices in Windows

Introduction

Dynamic Lighting aims to create cohesion across a fragmented ecosystem. This feature will provide rich end user experiences that will constantly delight users on Windows.

Dynamic Lighting provides Windows users and developers with native control of lighting devices implementing the open HID LampArray standard. By adopting an open standard, and by working with our OEM and ODM partners, Microsoft seeks to improve the RGB device and software ecosystem for users by increasing interoperability of devices and apps. Device manufacturers can use standardized firmware for the first time, enabling new native experiences across the Windows OS and apps without the high costs of proprietary firmware and software development.

Examples of experiences include synchronizing devices from different brands together in Windows Settings, applying effects intelligently across available devices, and leveraging app integrations to drive device lighting. For the first time, device manufacturers are empowered to focus purely on innovation and differentiation because their devices will be able to take advantage of a myriad of OS and app experiences.

This topic details which devices work with dynamic lighting, and provides documentation and samples so device manufacturers can build devices that are compatible with dynamic lighting.

Devices

HID LampArray devices are compatible with dynamic lighting. Devices are compatible via a native FW implementation, or through a VHF driver. The native FW implementation is also compatible with Xbox for keyboard and mouse devices.

Firmware developers can leverage the following samples to learn how to build devices that use LampArray:

- Arduino
- Macropad

Documentation on VHF drivers can be found at:

- Write a HID source driver by using Virtual HID Framework (VHF)
- HID Injector Driver sample

HID LampArray devices give their LED information to Windows, including position. This allows Windows to intelligently utilize device layouts for optimal effects. Additionally, there are several types of LampArray devices addressable by Windows:

- Keyboard
- Mouse
- Game controller
- Peripheral
- Scene
- Notification
- Chassis
- Wearable
- Furniture
- Art
- Headset
- Vendor-defined

Partners

Several partners have already joined the dynamic lighting effort:

- Acer is partnering with Microsoft on Dynamic Lighting. More details to come.
- ASUS ROG plans to introduce Dynamic Lighting compatible models later this year.
- Customers are seeking easier ways to manage RGB lighting within their gaming devices, which is why HP is planning to support Microsoft's Dynamic Lighting initiative across various OMEN, HP Victus, and HyperX products as an end-to-end launch collaborator.
- Logitech is partnering with Microsoft on Dynamic Lighting. All Logitech G LIGHTSYNC RGB devices are supported.
- Razer will have more Dynamic Lighting compatible products on an ongoing basis.
- SteelSeries is partnering with Microsoft on Dynamic Lighting. More details to come.
- Twinkly plans to release Dynamic Lighting-compatible devices.
- And several others!

Troubleshooting devices

- My updates are always a frame behind
  - This is likely a desynchronization issue between the LampArray device (ex. At 30Hz) and the app (ex. Also at 30Hz).
- The color I set is (slightly) different on different devices
  - Colors can vary slightly on devices from different manufacturers. This is somewhat expected.
  - This also applies to Brightness.

Ethernet

Article • 11/05/2020

Hardware

This topic covers recommendations for Ethernet in Windows 10. Due to potential mechanical design limitations, an Ethernet is considered optional on a tablet form factor. However, if implemented, the platform must provide a power gating option to reduce the power consumption of the integrated Ethernet chip.

Wake-On-Lan (WOL)

The default shutdown behavior puts the system into Hybrid Shutdown (S4) and all devices into the lowest power state (D3). In Hybrid Shutdown, user sessions are shut down while the contents of the kernel sessions are written to the disk, enabling faster startup.

WOL wakes the PC from a low power state when a network adapter detects a WOL event (typically, a specially constructed Ethernet packet). Remote wake from Hybrid Shutdown (S4) or Classic Shutdown (S5) is unsupported because Network Interface Cards (NICs) are explicitly not armed for wake in both the Classic (S5) and Hybrid Shutdown (S4) cases. Users expect zero power consumption and battery drain in both Shutdown states. This behavior removes the possibility of spurious wakes when explicit shutdown was requested.

In summary:

- Wake-On-LAN is only supported from Sleep (S3) or Hibernate (S4) states.
- Wake-On-LAN is not supported from Classic Shutdown (S5) or Hybrid Shutdown (S4) states.

Related topics

System Power Actions

USB Remote NDIS Devices and Windows

Graphics

Article • 11/05/2020

Windows 10 Guidelines

This topic covers guidelines for Graphics in Windows 10.

Windows 10 support for Ultra High Definition (UHD)

Windows supports 4K UHD and 8K UHD, but UHD is more than just higher resolution. These factors affect how people perceive images, in order of importance:

- Dynamic range, which is the difference between the darkest black and the whitest white, and the steps in between. UHD dynamic range allows users to see details in both the bright and dark portions of an image.
- Color gamut, which is the range of colors that can be faithfully recreated. UHD has a greatly expanded color gamut and bit depth.
- Frame rate, which is the number of frames per second. A higher frame rate (50/60fps) makes motion smoother, especially for sporting events.
- Resolution, which is the number of pixels. 4K resolution (2160p) has 4 times as many pixels as 1080p.

Where it makes sense, display panel manufacturers are recommended to invest in brighter panels, wider color gamuts, higher bit depths, and higher resolutions.

UHD investments in Windows 10

- HEVC decoder with hardware offload
- HEVC encoder with hardware offload
- Hardware DRM for PlayReady
- Support for wider color gamut
- Support for greater precision (color accuracy)

High Efficiency Video Coding (HEVC)

HEVC (also known as H.265) is a joint MPEG/ITU standard. It is non-proprietary and the successor to Advanced Video Coding (AVC, also known as H.264). Because it has significantly better compression, HEVC can consume up to 30-50% less bandwidth and

storage than AVC. It allows the transmission of high quality video across connections that have low bandwidth.

Content protection

Full 1080p, including 4K UHD, demands higher bar than existing software-based Digital Rights Management (DRM). Windows 10 supports hardware-based content protection. Hardware-based DRM is required to access premium content.

The infrastructure used for hardware-based content protection for Windows 10 is PlayReady 3.0, which provides these benefits:

- Hardware root of trust
- Device binding and all keys in hardware
- Compressed and uncompressed content not exposed to attack
- Output protections managed by hardware

Aspect ratios

- Windows 10 for desktop editions (Home, Pro, Enterprise, and Education) supports any aspect ratio, as in Windows 8.1
- Windows 10 Mobile will support any aspect ratio by Oct 2015
- Certain aspect ratios were targeted during design:
  - Phones: 16:9; Small tablets 16:10
  - 2-in-1's 3:2; Laptops/Desktops/TVs 16:9
  - Windows itself will reflow to support an arbitrary aspect ratio
  - "Square" aspect ratios (<4:3) may have limited scale factor support due to the 4:3 aspect ratio of the minimum effective resolution (800x600 for desktop, 480x800 for mobile)

Minimum requirements for Graphics

For details on minimum requirements for Graphics, see Minimum Hardware Requirements

Keyboard Japan - ImeOn / ImeOff Implementation

Article • 10/29/2020

Supported Windows Versions

The information in this topic applies to the following operating systems.

- Windows 10 Version 1903 (OS build 18362.1049 or later)
- 1909 (OS build 18363.1049 or later)
- 2004 (OS build 19041.388 or later)
- Versions of Windows 10 later than 2004

ImeOn / ImeOff key - Japan keyboard with Windows

This topic is for hardware keyboard manufacturers who plan to create ImeOn key / ImeOff key, or IME (Input Method Editor) developers who plan to provide the experience with the ImeOn key / ImeOff key running on Windows.

This topic provides a brief overview of ImeOn key / ImeOff key, guideline for HID usage, PS/2 Scan code and Virtual-Key code, and Microsoft IME behavior with the ImeOn key / ImeOff key on Windows.

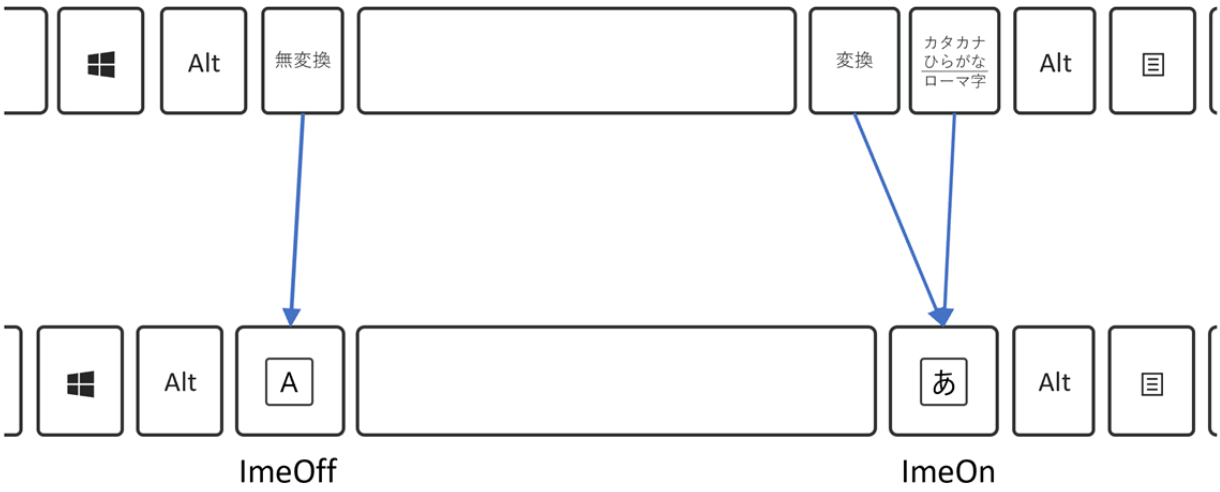
For keyboard manufacturers:

- Keyboard manufacturers must use the HID usage defined in HID usage, PS/2 Scan code, and Virtual-Key code for ImeOn key / ImeOff key section below for the usage the ImeOn key and ImeOff key send so that the usage can be transferred to the corresponding Virtual-Key codes in Windows.
- Keyboard manufacturers should use the same key top design described below to create a consistent hardware keyboard experience.

For IME developers:

- IME developers must define the experience on top of the Virtual-Key codes transferred from the HID usage the ImeOn key and ImeOff key send. See HID usage, PS/2 Scan code, and Virtual-Key code for ImeOn key / ImeOff key section for the Virtual-Key code values.
- IME developers should provide the same behavior of ImeOn key and ImeOff key as described in Microsoft IME behavior with ImeOn key - ImeOff key section for a consistent Japanese input experience.

ImeOn ImeOff key for a Japanese keyboard layout



- ImeOn key on the RIGHT of Space bar. Key top should be depicted as あ.
- ImeOff key on the LEFT of Space bar. Key top should be depicted as A.

Note: Eisu key (Caps Lock key) is expected to not print A, so as to avoid conflict with ImeOff key.

HID usage, PS/2 Scan code, and Virtual-Key code for ImeOn key / ImeOff key

The following table describes the HID Usage that ImeOn key / ImeOff key is required to send, PS/2 Scan code and Virtual-Key code translated from the HID Usage.

Key	HID Usage Name	HID Usage page	HID Usage ID	PS/2 Set 2 Make	PS/2 Set 2 Break	Virtual Key-code Constant/Value
ImeOn	Keyboard LANG1	0x07	0x90	0xF2	None	VK_IME_ON 0x16
ImeOff	Keyboard LANG2	0x07	0x91	0xF1	None	VK_IME_OFF 0x1A

For more information, see Virtual-Key Codes.

Microsoft IME behavior with ImeOn key - ImeOff key

Microsoft IME behavior with ImeOn key / ImeOff key is as follows for the specified states.

- **ImeOn key**

Key combination	No input/Converted strings	Only Input strings	Converted	Showing Candidate List	Changing POS boundary	Char Input into Converted POS
ImeOn	Enter Hiragana mode	Enter Hiragana mode	Enter Hiragana mode	Enter Hiragana mode	Enter Hiragana mode	Enter Hiragana mode
Alt + ImeOn (off by default)	Switch between Kana input and Romaji input	Switch between Kana input and Romaji input	Switch between Kana input and Romaji input	Switch between Kana input and Romaji input	Switch between Kana input and Romaji input	Switch between Kana input and Romaji input

Key combination	No input/Converted strings	Only Input strings	Converted	Showing Candidate List	Changing POS boundary	Char Input into Converted POS
Ctrl + ImeOn	Reconversion	None	None	None	None	None
Shift + ImeOn	Enter Full-width Katakana mode	Enter Full-width Katakana mode	Enter Full-width Katakana mode	Enter Full-width Katakana mode	Enter Full-width Katakana mode	Enter Full-width Katakana mode

- ImeOff key

Key combination	No input/Converted strings	Only Input strings	Converted	Showing Candidate List	Changing POS boundary	Char Input into Converted POS
ImeOff	Enter Half-width Alphanumeric mode	Enter Half-width Alphanumeric mode	Enter Half-width Alphanumeric mode	Enter Half-width Alphanumeric mode	Enter Half-width Alphanumeric mode	Enter Half-width Alphanumeric mode
Ctrl + ImeOff	None	Toggle Kana (1)	Toggle Kana(1)	Toggle Kana (1)	Toggle Kana (1)	Toggle Kana (1)
Shft + ImeOff	None	FullAlphanumeric (2)	FullAlphanumeric(2)	FullAlphanumeric (2)	FullAlphanumeric (2)	FullAlphanumeric (2)

(1) Toggle among Hiragana, Full-width Katakana, and Half-width Katakana.

(2) Toggle among un-capitalized, all-capitalized, and first-character-capitalized full-width alphanumeric characters.

Resources

Virtual-Key Codes

Using Keyboard Input

Keyboard Identifiers and Input Method Editors for Windows

Mobile Broadband

Article • 11/05/2020

This topic covers recommendations for Mobile Broadband components in devices that run Windows 10.

Hardware

Windows supports combinations of the GSM-based and CDMA-based mobile broadband technologies including LTE. The minimum connectivity capability for a Mobile Broadband device is 3G connectivity. 2G connectivity is optional. Devices for the various broadband technologies must follow the specifications of standards bodies and be certified by the preferred mobile operator.

- GSM/LTE
- CDMA

The following table summarizes the recommended mobile broadband hardware features.

Expand table

Feature	Remarks
Bus	USB-SSIC (USB 3.0) USB-HSIC for surface mounted solutions USB for connector mounted solutions (the device must continue to work while in USB selective suspend mode and capable of waking up the device)
Idle power	<10 mW (Connected Standby platform)
SIM Connector	User accessible

Embedded module solutions must always have mobile broadband functionality available without any USB hub design inside the chipset.

Idle mode power for 3GPP devices covers:

- PDP context activated but no network traffic state
- PDP context de-activated state

Idle mode power for 3GPP2 devices covers:

- Connected state with no network traffic state
- Disconnected state

Firmware

USB based devices for GSM and CDMA technologies (3GPP/3GPP2 standards based) must be firmware compliant with the Mobile Broadband Interface Model specification. The USB Forum must certify these drives for compliance (when that certification becomes available for mobile broadband devices).

The following table summarizes the additional features, as specified by NDIS, that we recommend the firmware support.

Expand table

Feature	Remarks
No pause on suspend	Support
Wake reason reporting	Support
USB SS	Support if USB-based
Radio Management	Support
Fast dormancy	Support

Drivers and software

Non-USB devices for GSM and CDMA use IHV developed Mobile broadband drivers. The drivers must comply with the Mobile Broadband Driver Model. IHV developed MB drivers must pass the tests for validating mobile broadband functionality, such as the Windows Hardware Lab Kit (HLK) tests.

- Devices based on the SPI peripheral bus need SPI driver support from the platform vendor. The SPI driver must comply with the features listed in Simple Peripheral Bus (SPB).
- USB devices must be compliant with the Microsoft USB driver stack and not require any IHV or third-party drivers for the mobile broadband functionality.

Multi-device

1. In case of MB+GNSS multi-function device, the GNSS driver can be an IHV driver as compliant with GNSS and power requirements. Please see GNSS requirements for more information.
2. No serial port devices enumerated for the purposes of diagnostics, device management or firmware upgrade etc. in the final production image.
3. Serial/AT command port, for the purposes of Mobile Network Operator (MNO) certification or OEM factory diagnostics/configuration should be supported through an IHV driver stack solution only. This IHV software stack / port drivers must be removed in the shipping image.
4. No IHV software based function SKUs to enumerate (or not enumerate) other functions in case of multi-function MB device solutions.
5. All functions, including any virtual functions (where needed) that are enumerated by the MB device need to be power-aware and respect Connected Standby and power states and cannot use legacy, non-power aware or non-PnP friendly solutions.

Connection Manager

We recommend using Connection Manager. There is no additional Connection Manager software necessary for the operation of mobile broadband devices. Value-add Mobile Broadband Connection Managers, if implemented, must implement the Mobile Broadband API.

The mobile broadband driver or device must implement the power management optimizations as required by NDIS.

Behavior in ACPI S3

If the platform supports ACPI S3 operating system power states, wireless devices shall not be removed from the bus upon entering S3.

Mechanical

SIM Card

If the device features a SIM card, we recommend the SIM card socket is accessible to the user to ease the activation process. If the device has a removable battery pack, the recommended location is inside the battery compartment.

Antenna

- In any direction, the gain of at least one antenna attached to the device should be better than -15dBi.
- The directive gain in any direction should not be higher than 6dB.
- The antenna efficiency should be better than 50%. Antenna efficiency includes losses due to impedance mismatch and the radiation efficiency.

Related topics

Mobile Broadband Interface Model (MBIM) specification

Windows.Devices.Sms namespace

Windows.Networking.NetworkOperators namespace

Mobile Broadband (MB) Design Guide

Mobile Broadband (MB) Reference

Overview of Mobile Broadband Windows Runtime API

Device App Lifecycle

Windows Hardware Compatibility Program

Near Field Communications

Article • 11/05/2020

Required, recommended, and optional near field proximity features

This topic covers recommendations for near field communications (NFC) in Windows 10. The following table lists required, recommended, and optional near field proximity features.

Feature	Remarks
Bus	Refer to Hardware Configurations. SPB API support using I^2C, SPI or UART
Antenna	Antenna dimensions can vary, as long as the effective actuation successfully connects at a range of 0 cm to 2 cm, is allowed but not required to connect at a range of 2 cm to 10 cm, and is prohibited to connect at a range greater than 10 cm (required). An antenna dimension of 40 mm x 60 mm is known to best achieve these results, but other configurations are acceptable as long as they meet the range of actuation requirements that enable the Windows scenarios (recommended). An antenna should be placed according to form factor along with the official NFC Forum N-Mark to indicate antenna location.
Implementation	Compliant with the guidance in NFC Design Guide (required).
Reliability	• [NFP] 95% success rate without errors based on 100 consecutive session attempts with 5 second intervals (required). • [Smart Card Reader] 95% success rate without errors based on 100 consecutive connection attempts with 5 second intervals (required). • [HCE] 95% success rate without errors based on 100 consecutive connection attempts with 5 second intervals (required). • [UICC] 95% success rate without errors based on 100 consecutive connection attempts with 5 second intervals (required).
Near field proximity	Near Field Communication defined by NFC Forum (required). A NFP provider must support the creation of sessions within 0.5 seconds from the point of being detected within the effective operating volume (required).
PC/SC Smart Card Reader	Required feature on all platforms.

Feature	Remarks
Radio Manager	Required feature on all platforms. Must implement the interface to match the defined commands from the Microsoft NFC Radio Manager.
Host Card Emulation (HCE)	Required on Windows 10 Mobile.
UICC Secure Element	This is an optional feature and is supported only on Windows 10 Mobile.

Hardware must comply with the Windows HLK requirements for near field proximity, including but not limited to accuracy, resolution, antenna placement, and range of values.

Firmware

Enumerate the device through ACPI.

Driver details

Near field proximity devices use IHV developed drivers. The drivers must comply with the guidance in Proximity Devices Design Guide. IHV developed drivers must pass the tests for validating near field proximity functionality such as the Windows HLK tests.

Mechanical

Antenna placement is critical for providing the best user experience and providing a consistent tap interaction between devices. Adding the NFC Forum's N-Mark on the device so the user knows where to tap devices together helps users discover and align the antenna, helping them complete the scenarios in the intended way. Additionally, Windows provides sound feedback during the Tap and Do experience.

The table below describes the location for the antenna by form factor.

Form Factor	Antenna Location and Considerations

Form Factor	Antenna Location and Considerations
Tablet	Place the NFC loop antenna near the surface of the device, not in the middle, with proper shielding to insure sufficient actuation volume (required). Place in the upper right-hand quadrant seen from the user holding the device to support the most natural interaction (recommended).
Convertible	Please use your best sense of judgement for convertible style systems.
Clamshell	Place the NFC loop antenna to the right of the touch pad, under the touch pad or under the right palm rest with proper shielding to ensure sufficient actuation volume (recommended).
All-in-One	Place the NFC loop antenna on the front of device (for example, bezel area) (recommended).
Desktop	If you are placing the NFC loop antenna on the chassis for a desktop, we recommend that the loop be placed on the top of the chassis near the edge. It is preferable to include the NFC chip in the keyboard or provide an external USB dongle.

Related topics

Simplifying wireless and network device discovery and pairing

Designing Systems and Developing Drivers for NFC

Connecting and Sharing with Near Field Communication

Proximity Devices

Supporting proximity and tapping (JavaScript)

Proximity and tapping (C#)

Proximity Devices Design Guide

Windows Hardware Compatibility Program

Minimum hardware requirements

NFC Devices

Peripheral Component Interface Express

Article • 03/22/2021

This topic provides recommendations for PCI Express (PCIe) in Windows 10. PCIe is a supported interface for form factors with devices requiring higher interconnect bandwidth. PCIe is most likely to be less energy efficient for battery-powered form factors compared to other mobile interconnect solutions. Chipset vendors and OEMs are advised to consider the overall power budget for the target device before selecting PCIe to connect a given peripheral chip.

Computers that run Windows Server 2016 must support PCI Express natively, and they must include a storage adapter and a network adapter that is compliant with the PCI Express architecture specification.

Related topics

Minimum hardware requirements

Sensors

Article • 05/12/2023

This article provides recommendations for the design and development of sensor devices. Windows provides native support for sensor devices. This support includes location sensors such as GPS devices. As part of this support, the platform provides a standard way for device manufacturers to expose sensor devices to software developers and consumers. The goal for this article is to enable ecosystem partners to build devices with optimized integrated sensor experiences with Microsoft technology.

Minimum hardware requirements and the Windows Hardware Compatibility Program

The minimum hardware requirements and Windows Hardware Compatibility Program requirements are fundamental for creating Windows-compatible audio solutions. Although the programs are optional, Microsoft recommends that audio products meet both sets of requirements to ensure basic audio quality.

For more detail on each:

- See Windows Hardware Compatibility Program

Sensor recommendations

The following sections cover recommendations for sensors. To ensure a high quality experience, all devices should be tested against these performance requirements.

Area	Type of guidance	Which devices should be tested
Device.Input.Sensor	Provides component-level guidelines in order to function optimally with the host OS in terms of software interfaces, communication protocols, and data formats.	All integrated sensors should be tested against these performance requirements.
System.Client.Sensor	Provides system-level guidelines in order to function optimally with the host OS in terms of software interfaces, communication protocols, and data formats.	All devices with integrated sensors should be tested against these guidelines.

Related topics

- Sensors Driver Design Guide
- Ambient Light Sensors
- Presence Sensors
- Integrating Motion and Orientation Sensors
- Sensors HID Usages
- Windows Hardware Compatibility Program
- Presence Sensing Device Experiences

Ambient light sensors

Article • 10/21/2022

This document provides recommendations for the design and development of devices with integrated ambient light sensors. Selecting a suitable ambient light sensor (ALS) device is critical.

The following high-level checklist is for developers integrating sensor hardware into devices. The rest of this document describes the process and background information in detail.

- Select a suitable backlight source.
- Select a suitable light sensor.
- Select optimized placement for the light sensor in the device enclosure.
- Perform per-model calibration taking into account all factors, such as coatings, light pipe, sensor configuration, placement, and so on. This should be performed using professional, pre-calibrated light meters.
- Integrate the sensors on to the device in one of the supported ways.
- Leverage the inbox HID Sensor class driver. Connect the device over USB, SPI, or I2C HID transport.
- Test the complete device as a light measurement instrument. Use various types of test lighting (incandescent, fluorescent, LED) at various intensities and compare values reported through the sensor platform with a high-quality light meter. The meter should measure light incident on the display of the device.
- Test the device and third-party drivers with the Windows Hardware Lab Kit (HLK) device requirements and related tests. Ensure that it runs correctly and passes all test cases.
- Ensure that OEMs, ODMs, and IHVs participate in mechanical design reviews for each major revision of device hardware.
- Ensure that sensor implementation is optimized mechanically, optically, and from an electrical engineering perspective.
- Test light sensors and adaptive brightness using the steps mentioned in Adaptive brightness test cases.

Integrating light sensors with device hardware

Several things can greatly affect what can be done with the information that the light sensors provide. These considerations include the following:

- The type of sensor, digital light sensors are preferred

- The accuracy, resolution, and field of view of the sensor
- The dynamic range of the sensor
- Infrared (IR) and ultraviolet (UV) rejection (human eye response)
- The supported bus technology (digital only)
- The digital sampling rate
- Power consumption
- Packaging and placement options

The following factors warrant special consideration:

- *Accuracy and resolution*: To deliver an optimal user experience for adaptive brightness and light-aware UI in applications, accurate sensor data is required as input. Generally, the more accurate the sensor, the better the corresponding user experience will be. A good goal for actual ambient light sensor (ALS) calibrated values is a consistent accuracy of within 4 percent of actual lighting conditions.
- *Dynamic range*: The dynamic range of a light sensor is the ratio between the largest and smallest values that the sensor can report, and defines range of lighting environments in which the sensor can be effective. A low dynamic range light sensor limits the environments in which it can be used. The ambient light sensors mounted on devices designed to be used outdoors, such as a cell phone, should support outdoor lighting conditions. Sunlight can range from 0 to 10,000 lux, or more. The ALS dynamic range for devices designed to be used indoors can be smaller. Indoor light typically ranges from 0 to 1000 lux.
- *Granularity*: To ensure the best experience the ALS should have a granularity of 1 lux when the ambient light is below 25 lux, and a granularity of 4% of the ambient light when it is above 25 lux. This enables the adaptive brightness algorithm to perform smooth screen brightness transitions.

The following are common lighting conditions to reference:

Lighting condition	**Illuminance (lux)**
Pitch black	1
Very dark	10
Dark indoors	50
Dim indoors	100
Normal indoors	300
Bright indoors	700
Dim outdoors (overcast)	1,000

Lighting condition	Illuminance (lux)
Sunlight outdoors	15,000
Direct Sunlight	100,000

Types of ambient light sensors

Ambient light sensors come in two fundamental types:

- **Analog light sensors** are connected to an embedded controller with an analog-to-digital (A/D) converter and require firmware that can accurately interpret the light sensor data and compensate for various conditions and phenomena that affect readings. Some examples of these phenomena include infrared (IR) light rejection and light frequency compensation. For example, fluorescent lights vary in intensity with the frequency of the AC power that is supplied to the fixture. Analog sensors are typically very inexpensive.
- **Digital light sensors** are more expensive than analog sensors but have advantages. Digital light sensors can automatically compensate for various conditions and phenomena. Digital sensors are also extremely compact. Some digital light sensors may provide coarse discrete lux measurements. The granularity of the readings in low light conditions must be carefully taken into account. Coarse, discreet measurements in low light conditions may result in a jarring brightness experience for the user.

Regardless of which type of light sensor is selected, accurate readings must be taken and exposed to the system.

Number of light sensors

The more ambient light sensors available to measure a lighting condition, the better the estimate of the actual illuminance. However, each light sensor adds cost and uses space on the device.

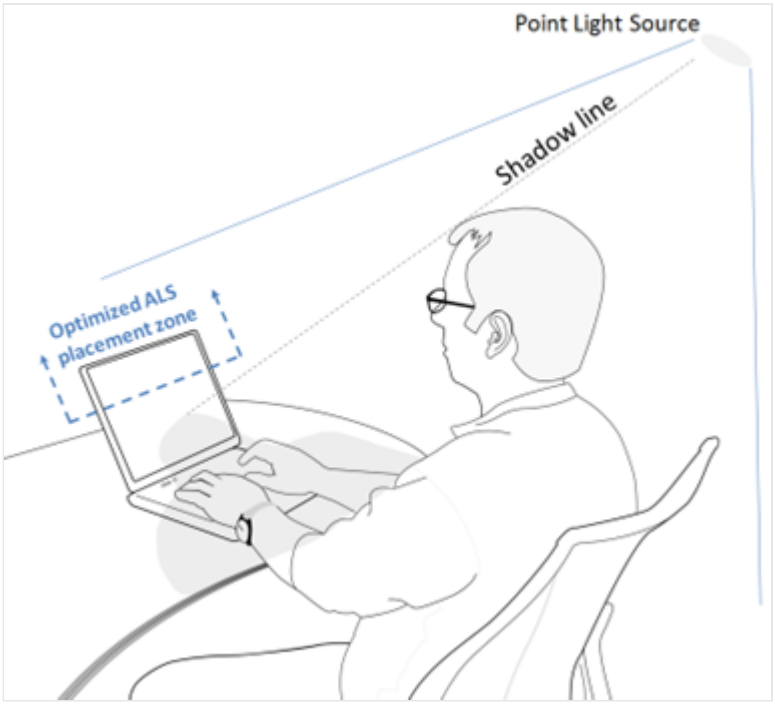
It's important that manufacturers strive for a solution that provides the system with the most accurate ambient light sensing capability. Inexpensive solutions may rely on a single sensor, but high end hardware can rely on a sensor array to provide the best possible measurement. If an OEM chooses to implement multiple ambient light sensors (to address issues like a hands or shadows obscuring the ALS), the OEM should expose one logical (consolidated) ALS to Windows and report the most accurate data.

If multiple sensors are exposed to the system, the single sensor used for automatic brightness should expose the DEVPKEY_SensorData_LightLevel_AutoBrightnessPreferred property. Similarly, OEMs may decide to fuse multiple ambient light sensors and expose the result as a virtual light sensor, also known as a pure software sensor. If both physical and virtual light sensors are exposed through the sensor device driver interface, the fused sensor must expose the DEVPKEY_SensorData_LightLevel_AutoBrightnessPreferred property.

Light sensor placement

Correct placement of light sensors is another critical aspect of good system design. The goal of the ALS is to measure the brightness of the environment as perceived by the user. The best theoretical sensor location would be between the eyes of the user. Real-world optimum placement of the light sensors is generally on the same plane as the display, facing the user. Sensors placed on the display have the advantage of detecting some glare that may occur on the screen.

Avoid placing the light sensor in areas of the computer that are likely to be obscured from light sources by shadows or a user's hands, fingers, or arm during normal usage. Figure below illustrates an example user scenario in which a direct light source is behind the user. A shadow is cast over the lower half of the screen and the base of the computer. This scenario suggests optimal light sensor placement near the top of the screen and facing the user.



Make sure the different configurations a device can take (position of the keyboard in tablet mode vs laptop mode for example) do not block the aperture and do not

intersect the sensor's field of view.

Finally, make sure the sensor's field of view isn't intersecting with any noisy source of light (camera flash, keyboard backlight, and so on) as these can contribute to additional noise or bad readings. Make sure to consider all the different configurations a device can take when considering the field of view intersecting with noisy sources of light.

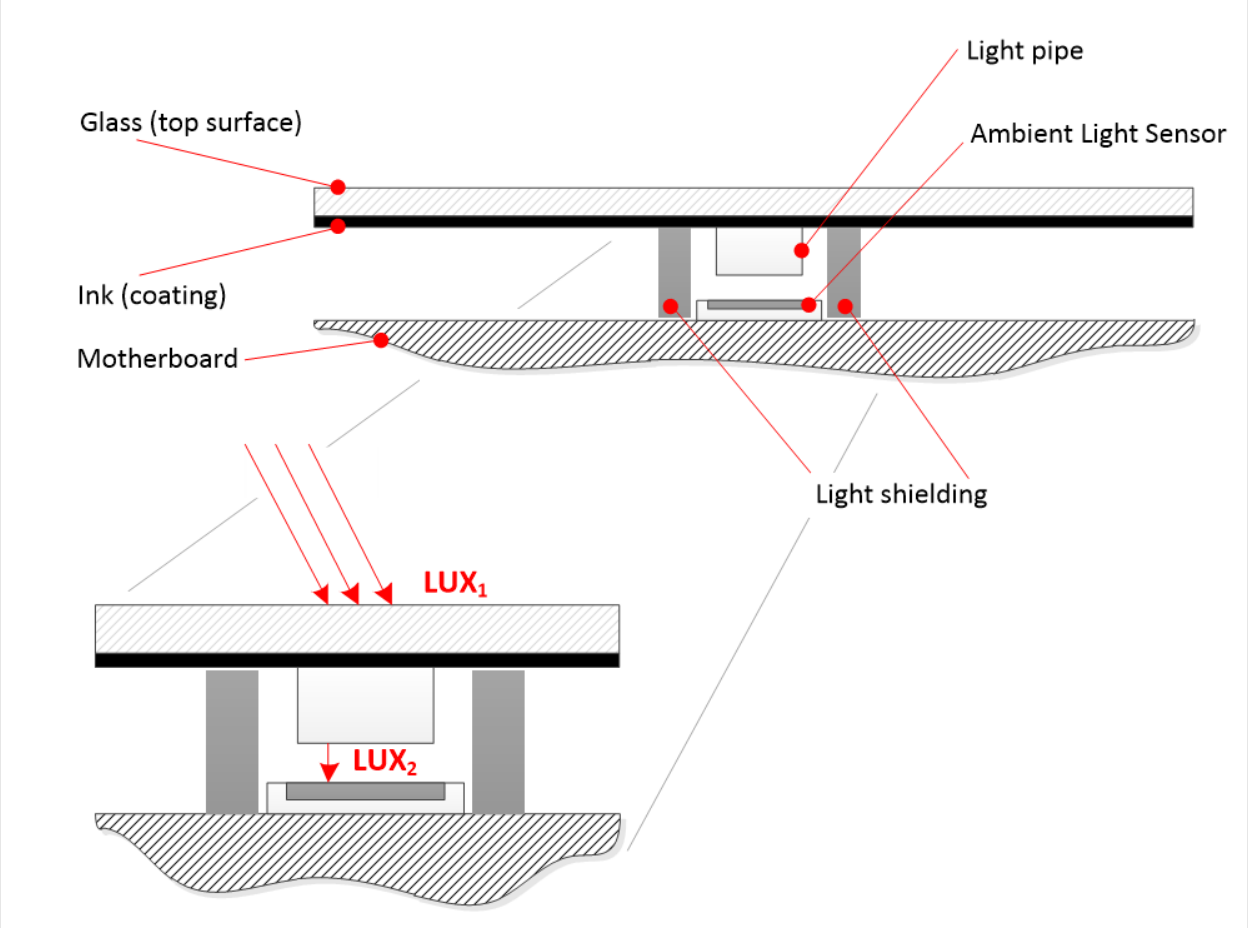
Dealing with invalid light sensor data

Under certain conditions, the ambient light sensor field of view may be obstructed by an object or by the user, making it impossible for the sensor to take an accurate reading. For example, such a condition may happen when the user's hand covers the ambient light sensor aperture. Many other cases exist.

The ambient light sensor can indicate this situation to the operating system by sending a new sensor sample with its PKEY_SensorData_IsValid data field set to FALSE. Proper hardware design should minimize the time and scenarios requiring this value to be set to FALSE as such a scenario prevents the system from properly controlling brightness. On an ideal system, the ambient light sensors would always be able to measure the ambient light, and this value would be set to TRUE.

Light sensor filters, lenses, enclosures, and calibration

When engineering a device that includes an ALS, the entire system of mechanical, optical, and electrical components related to the ALS needs to be carefully considered. The following diagram illustrates the key mechanical components that must be considered and understood when integrating and calibrating ambient light sensor hardware with Windows.



In this diagram, we see the following:

- Glass - outer surface of the screen
- Ink coating - black border around the screen
- Light shielding - prevents light bleed
- Light pipe - gathers and directs light to sensor
- Ambient light sensor
- Motherboard

> ⓘ **Note**
>
> Light pipes are usually not needed, and in many cases may degrade ALS performance. Please consult with the light sensor manufacturer for guidance regarding these kinds of optical components.

This diagram references two light levels:

1. LUX_{1}: The incident light level for the surroundings of the device at the surface of the display. This level is measured and reported by the ambient light sensor via the sensor platform.

2. LUX_{2}: The incident light level on the surface of the ALS. This is not the correct light level to report via the sensor platform as it does not take into account the attenuation factor of the optics.

The attenuation factor corresponds to how much light is blocked by the various components between the outer surface of the device (typically glass) and the sensing surface of the ALS. Attenuation can be calculated as follows: `A = (1 - transmittance)`

> **ⓘ Important**
>
> The ambient light sensor reports the ambient light intensity that it senses. Due to the transmissibility of the optics, the raw ALS readings report attenuated lux values and should not be used without correction. The transmissibility is the characteristics of the optics that reduce ambient light intensity and also reject infrared (IR) light. If the optics are painted with ink for visible appearance, an attenuation factor must be used to compensate for the corresponding reduction in ambient light intensity.

LUX_{2} should always be lower than LUX_{1}

The difference between these two lux values is called the attenuation factor. The attenuation factor accounts for the total percentage of light transmittance between the top surface of the glass (LUX_{1}) and the bare surface of the ambient light sensor (LUX_{2}). This is most drastic when a painted glass surface is used. The OEM, with the support of the sensor IHV, should measure the attenuation factor and correct for it in hardware before exposing the lux value to the operating system.

> **ⓘ Note**
>
> Transmittance is the ratio of the light level at the surface of the ALS divided by the ambient light level surrounding the device.

In the below example, assume the total percentage of light transmittance between the top surface of the glass and the bare surface of the ambient light sensor is 5%. In order to support the required lux range, the light sensor selected needs to support the following range at the bare sensor:

- $Minimum = 1 lux × 0.05 = 0.05 lux$
- $Maximum = 100,000 lux × 0.05 = 5000 lux$

In the firmware or driver, depending on whether a hardware or software ALS solution is being implemented, the following conversion is used to account for the attenuation factor:

$Output LUX = LUX_{1} = LUX_{2} / (total % _{light _transmittance})$

For a bare ambient light sensor reading of 100 lux, the following is the resultant output lux:

$Output LUX = 100 / 0.05 = 2000 LUX$

The entire system should be calibrated with proper light measurement equipment as well. This example only demonstrates the general considerations for part selection and initial calibration prior to formal calibration. Per-unit factory calibration is strongly encouraged for the best and most consistent user experience. Sensors often have accuracy ranges of +/- 20% from unit to unit, which can be accounted for via per-unit factory calibration.

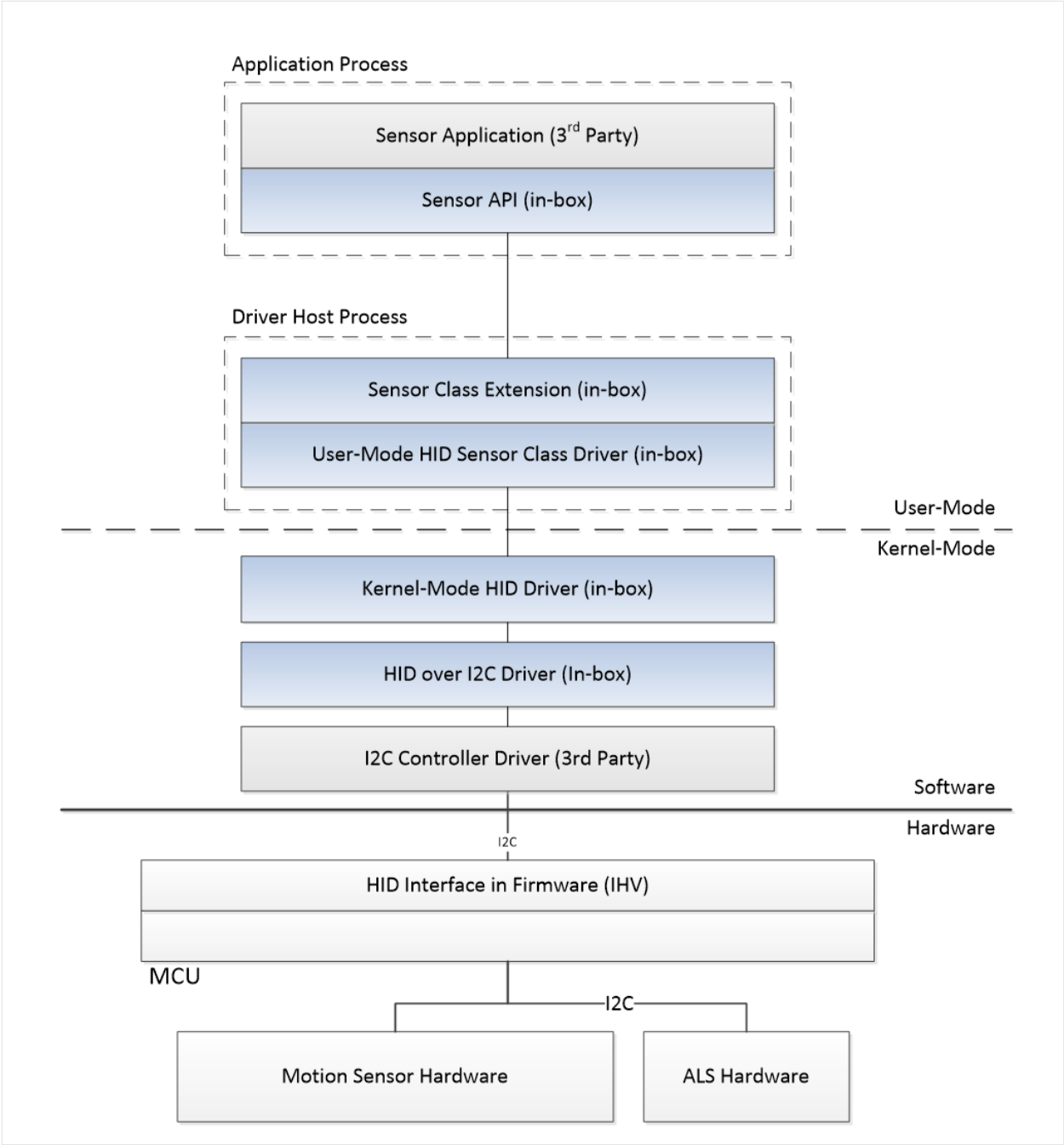
Also, the field of view is an important factor to consider in ambient light sensor placement and design. The smaller the field of view, the worse the performance of the sensor will be. As a general rule, a 55 degree half angle field of view (110 degrees total) is a fair target. The wider the field of view, the less prone the sensor will be to picking up a single point source of light or an area of shadow that may not accurately reflect the true light environment.

Sensor connectivity with HID and SPB

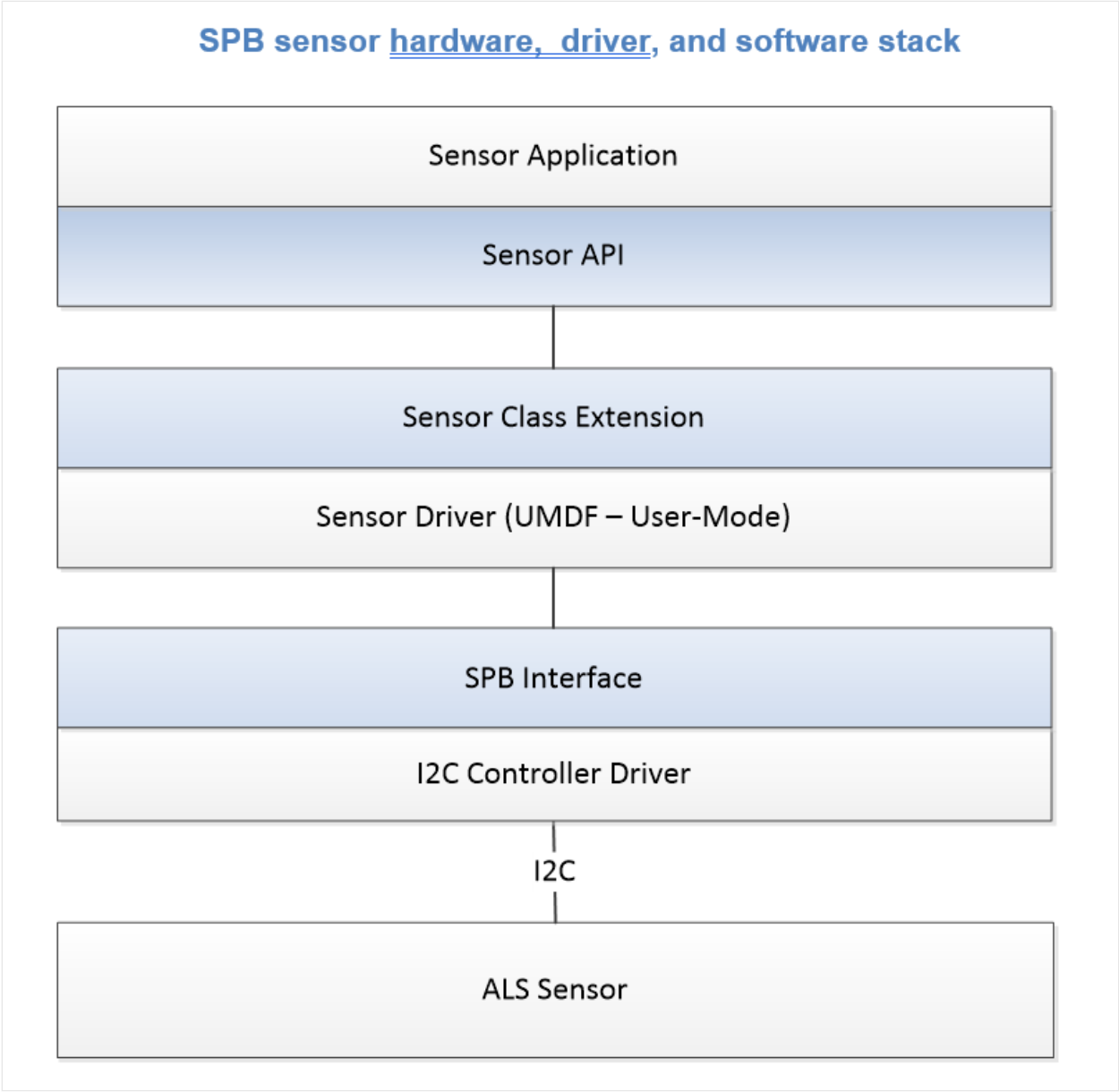
The following diagrams illustrate how to integrate ALS using the HID protocol, and with an IHV-specific driver for SPB.

> **Tip**
>
> The HID protocol is the recommend path to integrate ALS, making use of the inbox HID drivers in Windows.

HID sensor hardware, driver, and software stack is illustrated below:



SPB sensor hardware, driver, and software stack is illustrated below:



For more information about integrating sensor hardware via the HID protocol, including HID and I2C, please see Sensor HID class driver.

For more information about integrating sensors via SPB buses, please see the sensor combo sample driver source code on GitHub.

Ambient light sensor calibration

Professional calibration (recommended)

Calibration of the ALS in the integrated system using professional, pre-calibrated sensors in a controlled lighting environment is highly recommended. These pre-calibrated sensors, often called light meters, are available for purchase from electronic equipment vendors and online retailers.

Other calibration techniques

Details on other ALS monitoring and calibration tools are available through the Microsoft Ambient Light Tool article.

Light sensor validation

As a first step, you should always run the Sensors (i.e. Input) Hardware Lab Kit tests to validate the ambient light sensor. Ensure that all Minimum hardware requirements and the Windows Hardware Compatibility Program tests pass.

In order to validate proper functioning of the ambient light sensor:

- Make sure that the DisplayEnhancementService is started.
- Enable auto-brightness and set the slider to 50%.
- Validate that the display brightness changes when the lighting changes.
- Use a light dimmer to slowly ramp environmental light up and down and ensure that the lux values are *smoothly* ramping up and down. Coarse and discrete light changes result in sub-optimal screen brightness response and should be avoided.
- Use a professional lux meter to ensure that the ALS readings are accurate. At a minimum, verify the following points: 0, 10, 100, 500 and 1000 lux.
- In systems that only customized the ALR curve, test the behavior with users to validate ALR data meets user expectations.

Minimum hardware requirements and the Windows Hardware Compatibility Program

Minimum hardware requirements and Windows Hardware Compatibility Program requirements are fundamental for creating Windows-compatible sensor experiences. Although the programs are optional, we recommend that audio products meet both sets of requirements to ensure basic audio quality.

For more details, see the Windows Hardware Compatibility Program.

The following sections cover recommendations for sensors. To ensure high quality experiences, all devices should be tested against these performance requirements.

Area	Type of guidance	Which devices should be tested

Area	Type of guidance	Which devices should be tested
Device.Input.Sensor.AmbientLightSensor	Provides component-level guidelines in order to function optimally with the host OS in terms of software interfaces, communication protocols, and data formats.	All integrated ambient light sensors should be tested against these performance requirements.
System.Client.Sensor.AmbientLightSensor	Provides system-level guidelines in order to function optimally with the host OS in terms of software interfaces, communication protocols, and data formats.	All integrated ambient light sensors should be tested against these performance requirements.

Ambient response curve

If an ambient light sensor reports an ambient light response curve, then it must follow:

Data Field	Data Type	Definition
PKEY_LightSensor_ResponseCurve	VT_VECTOR	VT_UI4

The response curve of the sensor must have at least two points and the gradient must be positive or flat. For more information, see the response curve.

Color capable

Ambient light sensors are not required to detect color. If an ambient light sensor supports color, then the color related properties, thresholds, and data fields must be reported. A color capable light sensor must report the following enumeration property:

Data Field	Data Type	Definition
DEVPKEY_LightSensor_ColorCapable	VT_BOOL	Specifies if this light sensor is color capable.

A color capable light sensor must report one of the following combinations of data fields:

- Lux, kelvins, chromaticity x, chromaticity y
- Lux, chromaticity x, chromaticity y

For more information, see the light sensor data fields.

For the color data fields that the light sensor reports, thresholds must also be supported. A sample must be reported when at least one threshold is met. For more information, see the light sensor thresholds.

Data field properties

Ambient light sensors must report the required data field properties. For more information, please refer to light sensor data fields.

Data types

Ambient light sensors are required to report light data. For more information, see light sensor data fields.

Minimum report interval

Non-color-capable ambient light sensors in Windows are required to support a report interval of 250 milliseconds or less. Color capable ambient light sensors in Windows are required to support a report interval of 1000 milliseconds or less.

Thresholds

Light sensors are required to support thresholds on lux. If absolute thresholds are supported, both the percentage and absolute lux threshold must be met for a data sample to be reported.

Assume the absolute threshold is 1 lux and the percentage threshold is 25%:

Last sample	**Next sample**	**Result**
4 lux	3 lux	The next sample would get reported since the change is greater than or equal to 1 lux from the last reported sample and is greater than or equal to 25% of the last reported sample.
1 lux	0.5 lux	The next sample would *not* get reported since the change is less than 1 lux from the last reported sample.
100 lux	90 lux	The next sample would *not* get reported since the change is less than 25% of the last reported sample.

For more information, see light sensor thresholds

Auto brightness preferred

If an ambient light sensor is intended to be used with the auto-brightness feature, then the following enumeration property is required to be reported:

	Data Type	Definition
DEVPKEY_LightSensor_AutoBrightnessPreferred	VT_BOOL	Specifies if this light sensor should be the preferred light sensor used for the Windows auto-brightness service.

There must only one ambient light sensor reporting this property on a system.

Color calibration

Ambient light sensors are not required to support color. If an ambient light sensor does support color, then it is required to be properly calibrated.

While a light source is aimed directly at the sensor:

- The detected ambient lux is either within 10% or 1 lux of the actual incoming light
- The detected chromaticity x and y are within 0.025 of the actual incoming light

Enumeration properties

An ambient light sensor must report DEVPKEY_Sensor_ConnectionType even though this is not a required enumeration property for some other sensors

Is valid

Ambient light sensors are not required to report when light sensor samples are valid or not. If an ambient light sensor *does* support this, the following data field must be reported:

Data Field	Data Type	Definition
PKEY_SensorData_IsValid	VT_BOOL	Indicates if the current data sample is valid.

If the PKEY_SensorData_IsValid value changes, a sample must be reported regardless of if the thresholds have been met.

Assume the lux threshold is 1 lux:

Last sample	Next sample	Result
100 lux	100 lux, but the sensor is now blocked (previous sample's PKEY_SensorData_IsValid was true)	The current sample would get reported with 100 lux and PKEY_SensorData_IsValid set to false.
100 lux and was blocked (previous sample's PKEY_SensorData_IsValid was false)	100000 lux and the sensor is still blocked (PKEY_SensorData_IsValid is false)	No sample is reported.
0 lux and sensor was blocked (previous sample's PKEY_SensorData_IsValid was false)	0 lux, but the sensor is now unblocked (PKEY_SensorData_IsValid is true)	The current sample would get reported as 0 lux but with PKEY_SensorData_IsValid set to true.

Light calibration

The auto-brightness service in Windows needs light sensors to report an accurate measurement of the light level in the environment. When the light source is aimed directly at a light sensor that doesn't support color, the reported light level is required to be within 4%, or at least 1 lux of the actual incoming light level

Light range

The auto-brightness service in Windows needs to be able to detect a reasonable range of light levels from 1 to 10,000 lux. If the range is smaller than this, the adjusted auto-brightness may not be able to match the actual brightness of the environment.

See also

- Adaptive Brightness
- Sensors Driver Design Guide
- Sensors HID Usages

Presence sensors (human proximity)

Article • 05/12/2023

The following section details guidelines for OEMs selecting components such as presence sensors to support the inbox device experiences detailed above. It covers both the general requirements for all sensors in addition to the specific guidance for particular device experiences. Presence sensors are proximity sensors of type "Human". Review the device experiences and terms for Presence sensing before reading this section.

This section is intended to provide guidance for OEMs looking to install presence sensor hardware on a system. These guidelines represent a minimum viable sensor setup. Microsoft encourages installing hardware that outperforms the standards here.

The range, latency, distance, and power will not be measured by Microsoft but OEMs and ODMs should validate functionality in accordance with HLK requirements. If the system reports distance, its accuracy should be measured.

Important design details and general requirements for all human presence sensors

If a proximity sensor supports human presence, then it must support the detection type:

```
DEFINE_PROPERTYKEY(DEVPKEY_Sensor_HumanPresenceDetectionType,0xd4247382, 0x969d, 0x4f24, 0xbb, 0x14, 0xfb, 0x96, 0x71, 0x87, 0xb, 0xbf, 81); //[VT_UI4]
```

Currently defined detection types are found below:

Usage name	Usage type	Usage description
Human presence detection type: Vendor-defined non-biometric	Sel	Presence (of one or more people) is detected utilizing a vendor-defined, but non-biometric method. This is used to give positive affirmation that the sensor is using detection unrelated to biometrics as defined below. Without this, a Host cannot assume biometrics aren't utilized by the device.

Usage name	Usage type	Usage description
Human presence detection type: Vendor-defined biometric	Sel	Presence (of one or more people) is detected utilizing vendor-defined human biometrics. This is a catch-all for a Human Presence sensor that utilizes biometrics not already defined below.
Human presence detection type: Facial biometric	Sel	Human presence is detected by scanning (for example, by a low-resolution video camera) for human faces (for example, using Viola-Jones object detection). Distinguishing between faces or detection of facial attributes is not performed. Such detection is similar to that of existing digital cameras that can place a bounding-box around a face.
Human presence detection type: Audio biometric	Sel	Human presence is detected by scanning (for example, by a microphone) for human sounds (for example, a predefined keyword, general talking, loud noises, clapping). Distinguishing between voices or users, or detection of audio characteristics are not performed.

> **ⓘ Important**
>
> All Presence Sensors must accurately report the detection type described above.

Category 1 human presence sensor requirements

Measure	Metric	Comments
Sensor range	.2m – 1.2m *Note: This is a minimum viable range. Sensors with better ranges will be allowed.*	When directly measured (defined above), the user must be able to trigger a wake at 1.2m.
Sensor latency	<.33 second	This is the same as the Category II sensor latency. This requirement defines the time from firmware recognition of human presence to the operating system receiving that message. It is known that the operating system has variable latency time for wake once the signal is received from the sensor.

Measure	Metric	Comments
Sensor power - In Use	<80 mw for internal sensors; for external sensors power consumption is determined by power limitation of the bus; all measures average.	This measurement encapsulates the entire sensor subsystem, including the transmitter, receiver, and any LEDs used in accordance with the wake and lock scenarios only.
Sensor Power - Standby	<25 mw for internal sensors; for external sensors power consumption is determined by power limitation of the bus; all measures average.	Note this is the maximum peak power drain.
Accuracy – distance reporting	+/- 5 cm	Measured at 45 cm (12 in), 75 cm (29.5 in), and 120 cm (47.2 in). Only required if this sensor supports distance.

Category 2 human presence sensor requirements

Measure	Metric	Comments
Sensor range	.2m – 1.2m *Note: This is a minimum viable range. Sensors with better ranges will be allowed.*	When directly measured (defined above), the user must be able to trigger a wake at 1.2m.
Sensor latency	<.33 seconds	This requirement defines the time from firmware recognition of human presence to the HID Driver receiving that message. It is known that the operating system has variable latency time for wake once the signal is received from the sensor.
Sensor power - In use	<65 mw for internal sensors; for external sensors power consumption is determined by power limitation of the bus; all measures average.	This measurement encapsulates the entire sensor subsystem, including the transmitter, receiver, and any LEDs used in accordance with the wake and lock scenarios only.

Measure	Metric	Comments
Sensor power - Standby	<5 mw; for external sensors power consumption is determined by power limitation of the bus; all measures average.	This will be measured by the power drain of the sensor when the system is in Modern Standby. This measurement encapsulates the entire sensor subsystem, including the transmitter, receiver, and any other hardware used in accordance with the wake and lock scenarios only.
Accuracy – distance reporting	+/- 5 cm	Measured at 45 cm (12 in), 75 cm (29.5 in), and 120 cm (47.2 in). Only required if this sensor supports distance.

Device integration

Sensor placement is critical for providing the best user experience and providing a consistent Human Presence experience between devices.

The ideal placement is dependent on the form factor of the device, and in all cases OEMs should determine the best placement for their device. Additionally, for multi-posture devices like 2 in 1 convertible laptops, it is recommended that sensors firmware report the data field isValid = false for postures where Human Presence sensors will not produce accurate measurements (i.e. sensor is not facing user or is obstructed). The real-world optimum placement of the sensors is generally on the same plane as the display (facing the user).

Additionally, make sure the different configurations a device can take (position of the keyboard in tablet mode vs laptop mode for example) do not block the aperture and do not intersect the sensor's field of view.

Finally, make sure the sensor's field of view isn't intersecting with any noisy source of light (camera flash, keyboard backlight, and so on) as these can contribute to additional noise or bad readings. Make sure to consider all the different configurations the different configurations a device can take when considering field of view intersecting with noisy sources of light or electromagnetic waves.

Form factor	Human presence sensor location and considerations
Tablet	Place the human presence sensor near the surface of the device, not in the middle, with proper shielding to ensure accurate readings.
Convertible	Please use your best sense of judgement for convertible style systems.

Form factor	Human presence sensor location and considerations
Clamshell	Place the human presence sensor on the lid, above the display facing the user (recommended).
All-in-one or external monitor	Place the human presence sensor on the front of device (for example, bezel area) (recommended).
Desktop	If you are placing the human presence sensor on the chassis for a desktop, we recommend that it be placed on the top of the chassis near the edge. It is preferable to include the presence sensor in the monitor or provide an external USB dongle.

External human presence sensor support and arbitration policy

External human presence sensors integrated into monitors are supported by the inbox feature. The sections below detail the implementation, requirements, and policy that are applied to external human presence sensors. Other types of external sensors, such as standalone ones, are not fully supported because the form factor and usage type are too varied to reliably support. However, there is nothing explicitly prohibiting such sensors, and it is recommended vendors interested in producing such sensors reach out to sasensors@microsoft.com.

Implementation

External human presence sensors should be exposed to the OS identically to an internal sensor, except the DEVPKEY_Sensor_ConnectionType sensor property should be marked as external so the OS is aware of external connection. The OS will know if the sensor is integrated into an external monitor via the PLD information associated with the sensor. It is strongly recommended to expose the sensor as a HID device so the inbox HID class driver is used. This removes the need for the user to install drivers for the external sensor to become operational.

The physical connection between the external sensor and the system should be a USB connection separate from the display connection. This is because none of the standard display connection types, such as HDMI or DisplayPort, support HID. It may be possible for the display and the sensor to share one connection via USB-C alt-mode and USB4, but not all users will have access to such connecter types and, therefore, should not be relied on as the sole connector for the external sensor.

Requirements

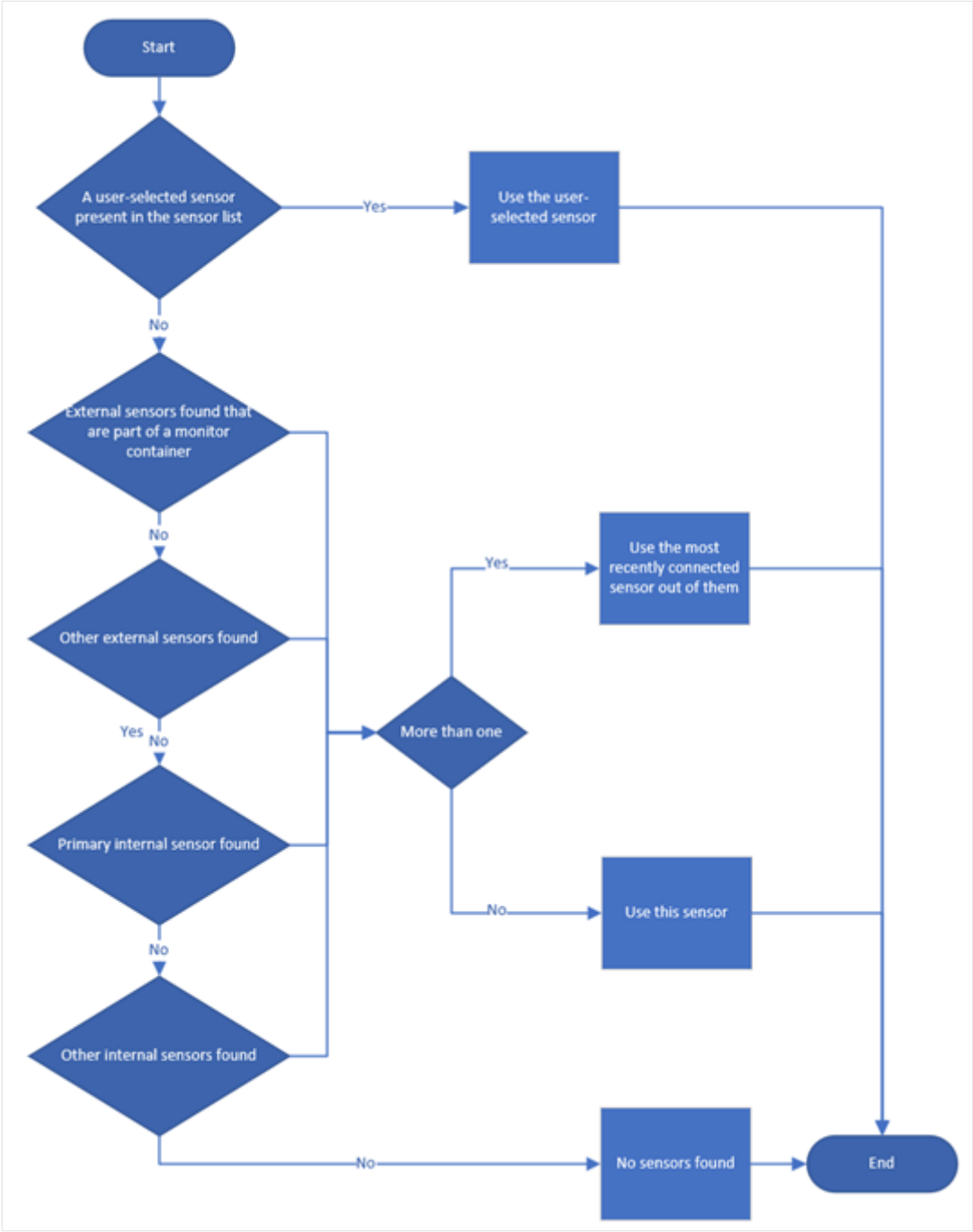
External human presence sensors must meet the same requirements as internal human presence sensors and pass the same HLK and manual tests.

Handling sensors with different capabilities

When there are multiple presence sensors with different capabilities, for example sensor A supports only wake on approach and lock on leave, while sensor B supports only adaptive dimming. The **Settings** page will reflect the capabilities of the selected sensor. If sensor A is selected only wake and lock toggles will appear. If sensor B is selected then only the adaptive dimming toggle will show.

Default presence sensor selection logic

Prior to and including Windows 11 22H2, the last connected sensor was chosen by default. Starting in May 2023 update to Windows 11 22H2 and newer, the selection logic is illustrated as below:



Policy for wake on approach, and lock on leave

External sensors have priority over internal ones and, when connected to the system, completely replace the internal as input to wake and lock. Priority means that when an external sensor is connected, it will become the default sensor for the system. The existing user preferences, such as timeout or detection distance, are transferred to the external sensor.

The arbitration policy for different external sensor configurations can be found below.

System has integrated sensor	System has external sensor	Arbitration policy
Yes	None	If the lid of the system is closed; wake and lock will be disabled and not function.
No	None	Feature is not present or is disabled.
Yes	Single	The externally connected sensor will become the default selected sensor for wake and lock. The user will have the option to configure which sensor they prefer in the Windows Settings user interface. This setting will only appear when there are multiple sensors available to the system. Following May 2023 update in version 22H2 or later, the selection logic follows default presence sensor selection logic.
No	Single	The externally connected sensor will become the default selected sensor for wake and lock. The user will have the option to configure which sensor they prefer in the Windows Settings user interface. This setting will only appear when there are multiple sensors available to the system. Following May 2023 update in version 22H2 or later, the selection logic follows default presence sensor selection logic.
Yes	Multiple	The same user option will be presented in the user interface. The sensor that was last connected to the system will be the default sensor until the user chooses another option. Following May 2023 update in version 22H2 or later, the selection logic follows default presence sensor selection logic.
No	Multiple	The same user option will be presented in the user interface. The sensor that was last connected to the system will be the default sensor until the user chooses another option. Following May 2023 update in version 22H2 or later, the selection logic follows default presence sensor selection logic.

Tip

Monitor orientation is not taken into consideration when determining if the sensor integrated on the monitor is usable, it is assumed sensors can still operate as normal in these cases.

Virtual human presence sensors

Human presence sensors must be backed by a physical device. In other words, a fake software device should not be exposed as a proxy to controlling wake, lock, or adaptive

dimming. This feature is only intended to be used with physical scenarios. The intention of this requirement is that Presence Sensors must detect and reference the physical environment to determine user presence signals. Virtual HID sensors may be implemented if they are aggregating data from other sensors like the microphone or other user inputs AND are running in an offloaded silicon such as an NPU or MCU where image or audio metadata is not accessible to applications on the OS. In the case of the camera, virtual presence sensors should NOT use or process image metadata in the OS. OEMs and IHVs who implement virtual sensors that consume image or metadata on the OS take on all legal liability for the privacy of such implementation.

Power requirements for presence sensors in this whitepaper are intended for physical sensors. For virtualized sensors that run part of other subsystems, the power requirements are inherited from the subsystem they are running in. For example a virtual presence sensor running on a Modern Standby system, and running in the offloaded Audio subsystem should follow the requirements listed in Audio subsystem power management for modern standby platforms

Interaction with camera privacy shutter and kill switches

It is expected that the privacy shutter does not interfere with human presence sensors unless image metadata is being transmitted to the OS. In cases where the same physical sensor is used with Windows Hello (IR camera) or the general RGB camera, then it is recommended that the human presence sensor is exposed via a separate physical path from the ISP (for example via a discrete NPU) where inferences and no image metadata (including facial signature) is transmitted to the OS. Since camera kill switches should operate in the ISP firmware, a physically separate path ensures that kill switches don't interfere with human presence features. If a separate physical path is not used for a shared sensor, then camera HLK guidance indicates privacy shutters operate on both the RGB and IR sensor and this would interfere with human presence features resulting in a poor user experience as Windows does not alert the user that human presence is blocked. It is not currently recommended to have a shared path between human presence and the ISP. However, in cases where this is necessary, ISP would stop HPD detection (for battery savings) and report that HPD sensor is not available.

It's highly recommended that any physical shutter report it's state to the OS via CT_PRIVACY_CONTROL (for UVC devices) or KSPROPERTY_CAMERACONTROL_PRIVACY (if using AVStream or DMFT driver). For details, see Camera Privacy Shutter Notification.

More details on the functionality of Camera Privacy shutters, kill switches and LEDs can be found at Camera Privacy Controls.

Interaction with camera privacy LEDs

It is Microsoft's expectation that camera privacy LEDs can and should remain off in cases where no image metadata is being transmitted to the OS. It is required that the human presence sensor is not physically able to transmit facial or image data to the OS. This can be accomplished with a variety of mechanisms such as but not limited to a dedicated bus or physically separate path from a camera ISP or a fuse within the sensor. More details on the functionality of camera privacy LEDs can be found in Camera Privacy Controls section.

Implementation guide

Readers of this section (who are intending to develop a presence sensor driver) should familiarize themselves with the Sensors Driver Design Guide.

In hardware a human presence sensor is implemented as a *SensorType_Proximity* with the `DEVPKEY_Sensor_ProximityType` set to `ProximityType_HumanProximity = 1`.

If the sensor utilizes biometrics it can be in the *SensorCategory_Biometric*.

Device bus connectivity

There are no hard requirements outside of the tests in the HLK (see below) on device bus connectivity. However it recommend that system builders leverage inbox drivers available in Windows 11 where possible. For example, there is an inbox HID driver for I2C, and an inbox Sensor Driver for HID-based devices. This offers the benefit that Microsoft will maintain inbox drivers in Windows 11.

Power management

This section contains a basic overview of the power management of human presence sensors. For more detailed information see Sensors Power Management.

> **Tip**
>
> The presence sensor is designed to function on modern standby systems. Additional support for wake on approach for non-modern standby systems has been added in the May 2023 Update to Windows 11 Version 22H2 and later.

Additional requirements for HIDUSB implementations

The Selective suspend for HID over USB devices article gives an example of how an OEM or IHV INF references the inbox INPUT.INF to enable a different selective suspend feature. OEMs and IHVs may do similar in their INF but referencing the section mentioned above to enable wake-on-touch.

To simplify this INF, OEMs and IHVs may also consider creating an extension INF instead. Please refer to the Using an Extension INF file documentation.

If the device has multiple HID collections including human presence sensor collections, non-human presence collections, the sensor firmware should only support wake from the human proximity collections. If a non-human presence sensor collection may signal wake too, the device will be still brought back to D0 to drain the power.

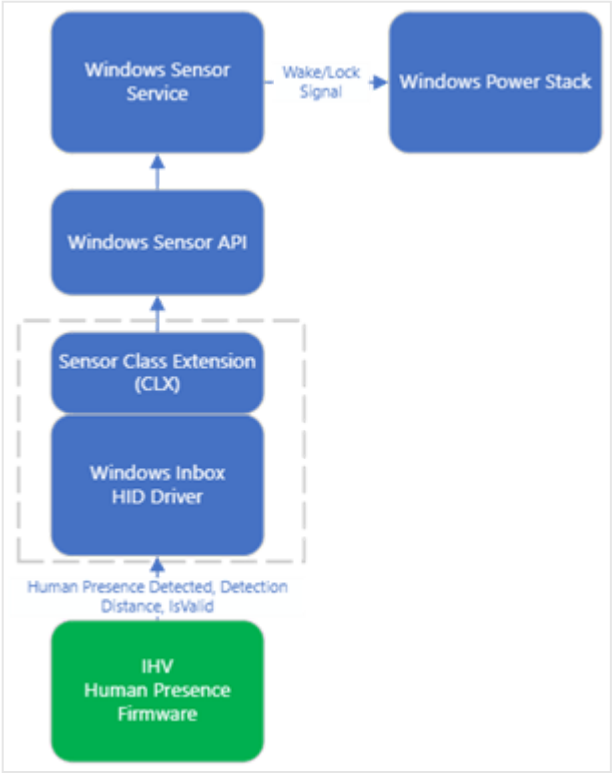
Protocol implementation (high-level architecture)

This section requires the necessary driver implementation details to report a human presence sensor to the OS. Technically this is done by implementing a human proximity detection Type for a proximity type sensor in the biometric sensor category.

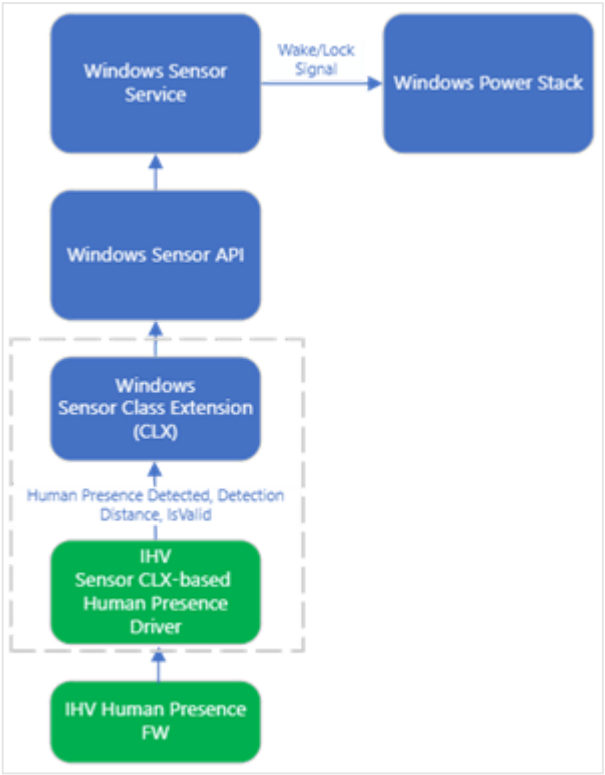
The diagrams below represent how the data from the sensor flows up the stack when the feature is functioning. External monitors will use the diagrams that are labeled as "HID".

Driver Type / Sensor Capability	**Distance-Capable HW (+/- 5cm)**	**Distance-Agnostic HW**
HID: Human Interface Device	HID-based distance-capable data flow (recommended)	HID-based distance-agnostic data flow
CLX: Windows sensor class extension	CLX-based distance-capable data flow	CLX-based distance-agnostic data flow

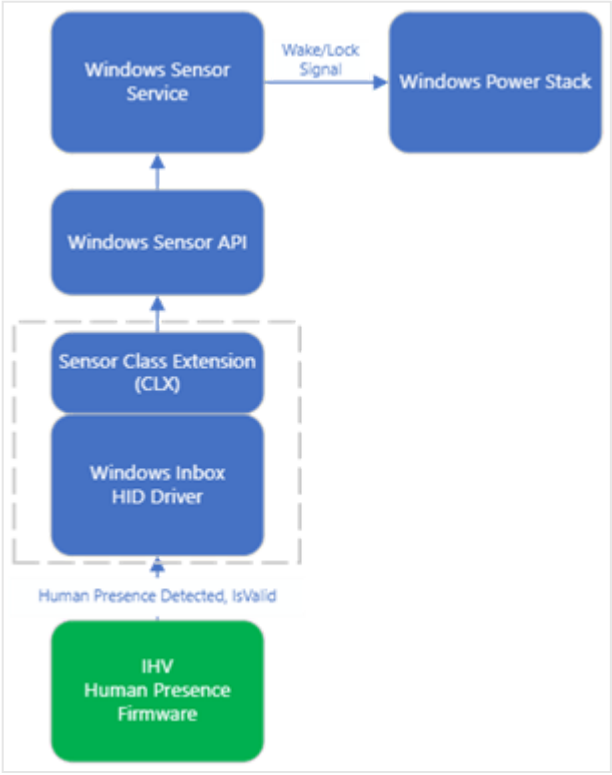
HID-based distance-capable data flow (recommended)



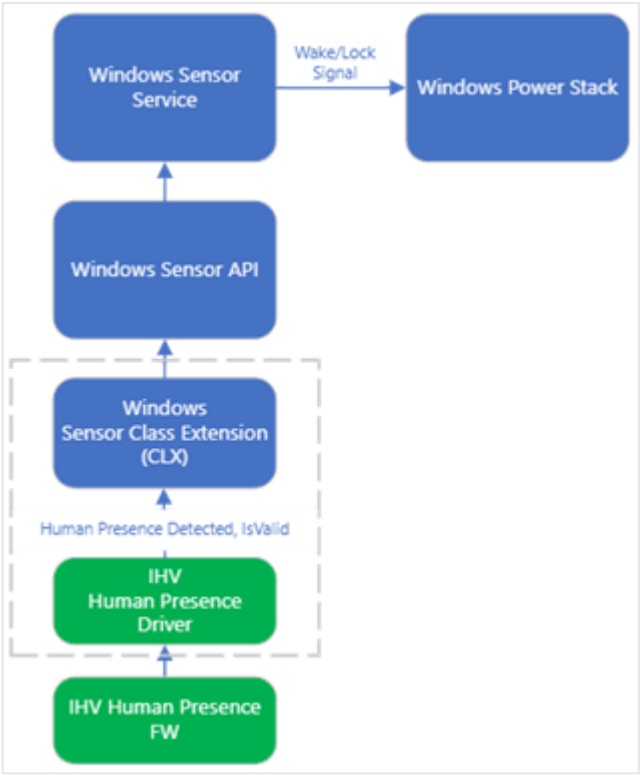
CLX-based distance-capable data flow


Windows Sensor Service
Wake/Lock Signal
Windows Power Stack
Windows Sensor API
Windows Sensor Class Extension (CLX)
Human Presence Detected, Detection Distance, IsValid
IHV Sensor CLX-based Human Presence Driver
IHV Human Presence FW

HID-based distance-agnostic data flow



CLX-based distance-agnostic data flow



High level data flow

1. User modifies the wake or lock settings via the settings UI and is communicated to the Windows sensor service.
2. Windows sensor service starts the human presence sensor during normal operation or modern standby to monitor for wake and lock.

3. Human presence sensor raises a new sample which contains human present, detection distance, and is valid.
4. If the system is in modern standby and the new human presence sample reports a human within the configured detection range the Windows sensor service will send a wake signal to the Windows power stack. Conversely, if the system is in normal operation and a human is not reported or is outside of the configured detection range the Windows sensor service will send a timeout signal to the Windows power stack.

Human presence signals

Signals:

- **User present signal** – Occurs when the sensor reports a data sample with human present data field set to true. If system supports attention, then attention is set to engaged/true in this state.
- **User present not engaged signal** – Occurs in systems that support attention when the sensor reports a data sample with the human present data field set to true and then attention is set to unengaged or false.
- **User not present signal** – Occurs when the sensor reports a data sample with the human present data field set to false.
- **User unknown signal** – Signaled when human presence is unknown. This can occur if the sensor is spuriously removed or if the sensor has yet to report a valid sample.

Required HID descriptors and top-level collections

If implemented, using one of the HID-based architectures the following section describes the data fields for exposing presence information.

For additional information on exposing a device through HID see Sensors HID Usages.

Human presence – present (wake and lock)

Usage ID	Comments
0x04B1 HID_USAGE_SENSOR_DATA_BIOMETRIC_HUMAN_PRESENCE	Usage to expose a Boolean that indicates presence. Set this value to "1" to indicate human presence. Set this value to "0" to indicate no human presence.

Usage ID	Comments
0x04B2 HID_USAGE_SENSOR_DATA_BIOMETRIC_HUMAN_PROXIMITY_RANGE	Used to expose distance values in the input report Default unit is in meters but the sample report descriptor in this document shows how the device can report millimeters. This will be made an optional field in a future Windows update.

> **Note**
>
> The millimeter report must be accurate to within +/- 5000mm. The reported range may be continuous or in discrete increments of less than 5000mm (i.e. 2000mm, 7000mm,..).This will be made an optional field in a future Windows update.

Validation guide - Minimum requirements and testing: Windows Hardware Certification Program (WHCP)

The minimum test and requirements to certify your hardware as compatible is outlined in the Hardware Lab Kit (HLK) tests in the Windows Hardware Compatibility Program Specifications and Policies.

Related topics

- Presence sensing device experiences
- Presence sensor API documentation
- Sensors driver design guide
- Sensors HID usages
- Windows Hardware Compatibility Program

Simple Peripheral Bus (SPB)

Article • 06/21/2022

> **Note**
>
> This article contains references to the term *slave*, a term that Microsoft no longer uses. When the term is removed from the software, we'll remove it from this article.

This topic covers recommendations for Simple Peripheral Bus in Windows 10.Windows includes support for low-power, simple buses, such as Inter Integrated Circuit (I^2C) and (I^2C) and Simple Peripheral Interface (SPI), using framework extensions of the Kernel Mode Driver Framework (KMDF) architecture. Controller drivers are not provided in-box. Chipset vendors, OEMs, or IHVs must develop a controller driver implemented in KMDF. The architecture provides flexible device configuration topologies supporting simultaneous use of buses for control and data transactions, as well as GPIO for signaling and interrupts. The complete device definition is defined through Advanced Configuration and Power Interface (ACPI).

In Windows, buses are supported through KMDF controller drivers. With the aid of the KMDF platform, the controller driver is used predominantly to define the hardware-specific interfaces necessary to enable controller function.

The Windows infrastructure supports devices that share buses, buses that are multiplexed on the same line, and device configuration through ACPI. Windows uses ACPI as the primary means for device identification, configuration, and control.

The following table summarizes the support for the Simple Peripheral Bus.

Bus	Inbox Support	Framework extension provided	Third party allowed	Additional support details
I^2C	No	Yes	Yes, using SPB Framework extension	Master only "General Call" is not supported Direct Memory Access (DMA) supported

Bus	Inbox Support	Framework extension provided	Third party allowed	Additional support details
SPI	No	Yes	Yes, using SPB Framework extension	Master only, "General Call" is not supported Full duplex supported DMA supported
MIPI-HSI	No	No	Yes, using Windows Driver Foundation (WDF)	
MIPI-SLIMbus	No	No	Yes, using WDF	
MIPI-CSI	No	No	Yes, using WDF	
UART	No	Yes	Yes, using Serial Framework extension (SerCx2)	DMA supported Custom transfer modes supported with SerCx2

Design considerations for SPB

The following are some generic considerations for SPB:

- SPB is not a Plug and Play bus. Peripheral devices typically have fixed connections to an SPB and cannot be removed. System manufacturers must ensure accurate information in ACPI to enumerate the SPB-connected peripheral devices for the Plug and Play manager, and specifies the hardware resources that are dedicated to each device.
- There is no in-band interrupt support for SPB. Most peripherals support device signaling through a separate interrupt (often GPIO based) mechanism, and accurately mapped in ACPI.
- Windows provides support for the SPB Class Extension (spbcx.sys) in Windows 8 and beyond. SoC partners are responsible for developing and redistributing their appropriate SPB Controller driver.
- Peripheral drivers for SPB devices are generally provided by SPB Device partners. Microsoft provides one class driver for SPB devices for HID over I^2C (hidi2c.sys).

- Device classes may provide HLK requirements or WEG guidance around the following topics related to I²C:
  - Sharing I²C controller with other devices
  - Preferred I²C signaling speed
  - Power management and wake scenarios over I²C and GPIO.

- Inter Integrated Circuit (I²C): I²C is the primary bus that is validated as part of SPB and is highly recommended on SoC systems.

  Microsoft provides Windows Hardware Compatibility Program requirements for I²C. Use the Windows Hardware Lab Kit (HLK) to test devices against those requirements.

- Simple Peripheral Interface (SPI): Support for SPI is optional and up to the SoC partner. The Windows Hardware Compatibility Program does not contain any requirements specific to SPI bus.

Support for SPB across systems

Microsoft supports SPB on Arm systems and x86/x64 platforms (running in S3 configurations). Microsoft supports SPB on platforms running in both Connected Standby (CS) and S3 configurations.

Please contact your platform provider for drivers and support.

There are a number of device scenarios that leverage SPB for connectivity. I²C is available on CS and S3 traditional power model. Modern SoCs with on-SoC sensor low power cores can implement non-I²C solutions as needed.

Devices on removable docks/ports should also follow the guidance around docking scenarios, also included in the WEG. Some of those devices may make more sense over buses like USB rather than I²C.

SPB framework extension

The SPB framework extension library extends the Windows Driver Framework to support SPB drivers. The SPB framework simplifies development of an SPB controller driver and improves the compatibility between peripheral drivers and the controller driver by providing common implementation of the "top-half" of the driver that processes I/O requests (as compared to the "bottom-half", which is driven by the top-half and controls the hardware). The SPB Framework Extension is a KMDF extension library. It handles the up-front processing of SPB request and the sequence in which they are handed to the

controller driver. The SPB framework extension is designed to support I²C and SPI buses, and may be appropriate for other buses with similar semantics.

Serial Framework extension

The serial framework extension library extends the Windows driver framework to support serial controller drivers. Similarly to the SPB framework, the serial framework simplifies development of a serial controller driver and improves the compatibility between peripheral drivers and the controller driver by providing common implementation of the "top-half" of the driver that processes I/O requests. The serial framework extension is a KMDF extension library. It handles the up-front processing of the calls to the serial APIs and the sequence in which they are handed to the controller driver. The serial framework extension is designed to support the modern UART controllers and simplify controller driver implementation and diagnosability.

I²C and UART HLK requirements

There are Hardware Compatibility Program requirements for I²C and UART controllers. Requirements for SPI are being considered for the future as well. The logo requirements are primarily intended for SoC silicon vendors for the bus interface hardware and the associated controller drivers. OEMs and ODMs are not required to revalidate the hardware or controller driver but are welcome to run the tests if desired. Special set-up steps are required to validate these requirements. The setup includes the following:

- An open system with accessible I²C /UART pins/ports
- Modifications in ACPI to expose the I²C/ UART test device to software
- A specific test device (WITT) attached to the system under validation

For additional set-up information, please refer to the Hardware Lab Kit (HLK) documentation.

Peripheral drivers

Peripherals are enumerated by ACPI and are generally static. Peripheral function drivers determine their appropriate bus resources by interacting with the framework extensions. Peripherals and controllers are not hierarchical, and peripherals may use several SPB, GPIO, Serial, and other high speed buses. Peripheral drivers that access embedded devices, such as sensors, input devices, modems, and radios, may be written in kernel mode or user mode. These drivers may be portable across different ODM or OEM board configurations as long as ACPI is updated appropriately.

Firmware

Controller ACPI settings and bus parameters are vendor-specific and dependent on the particular controller. The following table summarizes the ACPI settings for the controller and the peripheral bus.

Bus	Controller ACPI settings	Peripheral ACPI settings
I^2C	Controller addresses	Bus address
	Pin configuration	Clock rate
		Slave mode
		Addressing mode
SPI	Controller addresses	Chip select line
	Pin configuration	Clock rate
		Clock polarity
		Clock phase
		Wire mode
		Device selection
		Device selection polarity
		Slave mode
UART	Controller address / pin	Initial baud rate
	Configure initial baud rate	Parity
		Start bit and stop bit length
		Flow control method(Hardware/Software/None)
		Lines in use
		Receive buffer size
		Transmit buffer size
		Endian-ness

For an ACPI example, see How to Get the Connection Settings for a Device. For more information about how to integrate devices on low power buses, see Accessing GPIO, I2C, and UART Devices.

Tools and technical reference

Resource Title	Content type	Description	Link
Using the Windows Driver Framework to build better drivers	Video	Discusses how the WDF can improve driver reliability and how to better realize power-savings and deploy drivers on multiple versions of Windows.	Channel 9
Understanding Low-Power Buses	Video	Demonstrates how to integrate a device on the new buses and create a driver. You will learn how to write ACPI to enumerate your peripheral and get started writing and testing a peripheral driver.	Channel 9
Kernel-Mode Driver Framework Design Guide	Article	Introduces Kernel-Mode Driver Framework (KMDF).	MSDN
UMDF 1.x Design Guide	Article	Introduces User-Mode Driver Framework (UMDF).	MSDN
Windows Hardware Compatibility Program	Article	Provides information on the Windows Certification Program.	MSDN

Trusted Platform Module (TPM)

Article • 11/05/2020

This topic provides recommendations for Trusted Platform Module (TPM) technology for Windows 10.

- Minimum TPM requirements - see Minimum hardware requirements
- TPM recommendations (TechNET)

Windows Digitizer Class Input Devices

Article • 01/18/2019

Touch interfaces can be found today across myriad devices ranging from mobile phones to slates, to kiosks, to horizontal/vertical displays of 30 inches. This section presents implementation and validation guides for the various Windows touch technologies.

In this section

Expand table

Topic	Description
Pen Devices	This section provides information about how to implement and validate an integrated Windows pen device, for Windows 10 and later operating systems.
Precision Touchpad Devices	This section provides information about how to implement and validate a Windows Precision Touchpad device, for Windows 10 and later operating systems.
Touchscreen Devices	This section provides information about how to implement and validate an integrated Windows Touchscreen device, for Windows 10 and later operating systems.

Pen Devices

Article • 01/18/2019

This section provides information about how to implement and validate an integrated Windows pen device, for Windows 10 and later operating systems.

In this section

Topic	Description
Pen Implementation Guide	This section is an implementation guide for an integrated Pen device, for Windows 10 and later operating systems.
Pen Validation Guide	This section is a validation guide for an integrated Pen device (a Windows Active Pen), for Windows 10 and later operating systems.

Pen Implementation Guide

Article • 05/20/2024

This section is an implementation guide for an integrated Pen device, for Windows 10 and later operating systems.

The information includes guidance on using the Human Interface Device (HID) protocol to communicate with a Windows host.

In this section

Expand table

Topic	Description
Windows Pen Designs	This topic presents information about the pen devices that are compatible with Windows 10.
Device Bus Connectivity	This topic discusses bus connectivity methods for a Windows pen device.
Power Management	This topic provides information about pen power management, and discusses power consumption requirements.
Pen Protocol Implementation	This section provides guidelines for pen protocol implementation. Integrated Windows pen devices are expected to use the Human Interface Device (HID) protocol to communicate with the host.
Haptic Pen Implementation Guide	This topic provides implementation details for haptic pens.
Digitizer Display Mapping	This section provides guidelines for successfully mapping digitizers to their display.

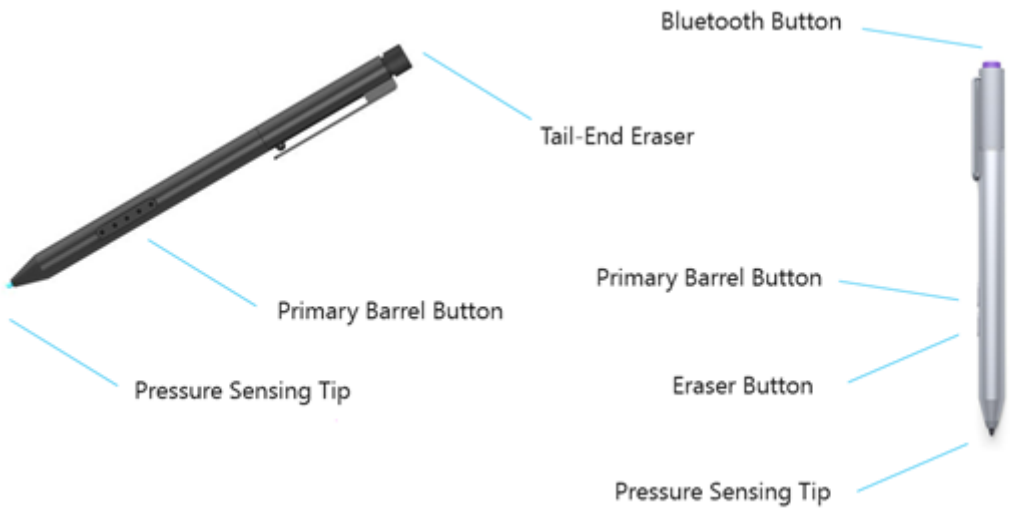
Windows Pen Designs

Article • 05/09/2023

This topic presents information about the pen devices that are compatible with Windows 10.

Physical Design

The physical pen device that the user interacts with may have a variety of industrial designs. However all compatible integrated Windows 10 pen devices have a pressure sensitive tip and an erase *affordance*. The erase affordance can be implemented as a physical button on the pen, or as a tail-end eraser (similar to a traditional pencil). The following image illustrates these potential implementations; a pen with a tail-end eraser on the left and a pen with an eraser button on the right.



For information about the erase affordance, the barrel button, and how to report pressure, see Windows Pen States.

Button Placement

If implementing a Bluetooth button to support the quick launch features of Windows Ink Workspace, please ensure that the button is placed at the tail-end of the pen device as shown in the above figure.

> **Note**
>
> Bluetooth buttons should not be placed on the side of the barrel, nor should barrel buttons utilized for selection or erase be overloaded with functionality that

depends on the pen's position/state, as this typically leads to a subpar user experience.

Bluetooth Button Implementation

To implement a tail-end Bluetooth button, the device will report 3 distinct keyboard combinations corresponding to 3 distinct button actions via a HID Bluetooth LE keyboard device exposed to the host. The actions and corresponding keyboard combinations are outlined below:

Bluetooth Button Action	Key Combination to Report
Single-click	WIN+F20
Double-click	WIN+F19
Press and hold	WIN+F18

Pen Stowage

Starting with Windows 10, version 1903, Windows supports notifications for devices incorporating compatible pen stowage. The mechanism relies on the hardware detecting the pen being removed or replaced and generating a corresponding HID keyboard report for a pair of shortcut combinations. To signal a dock (pen replaced in stowage), report **WIN+CTRL+F20** and to signal an undock (pen removed from stowage), report **WIN+CTRL+F19**. This can be implemented with firmware or a driver.

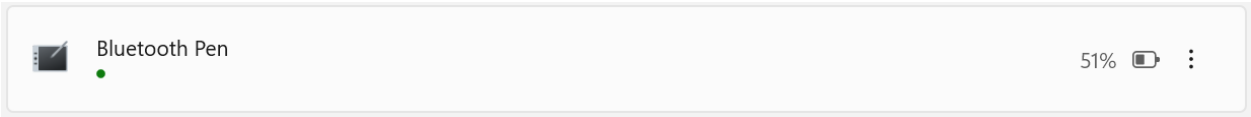
These undock/dock events bring-up/dismiss the Shell ink workspace menu. Starting with Windows 10, version 2004 Office also reacts to these events by using a platform API that enables any developer to extend their application for awareness of stowage events. There is no support to query whether the pen is present in the dock, apps are only notified of removal and return events if they are in the foreground.

Reporting Battery Level

A pen's battery level may be reported via Bluetooth LE Battery Service 0x180F and using Battery Level (0-100%) 0x2A19 for paired devices (keyboards, mice, pens, external PTPs, radial controllers, etc.)

When reported via the above, Windows will communicate the current battery level to users in two ways.

First, users can view the current battery level by going to the **Bluetooth & devices** > **Devices** settings page. An example pen entry is shown below:



In addition, when a pen's battery drops to 5% or lower, a Windows notification will be shown to the user indicating that the battery is low. Similary, this same notification will be shown if a pen's battery is at or below 5% battery when first paired.

> **Note**
>
> Users may disable the low-battery notification, in which case they must go to the **Bluetooth & devices** > **Devices** settings page to view the battery level.

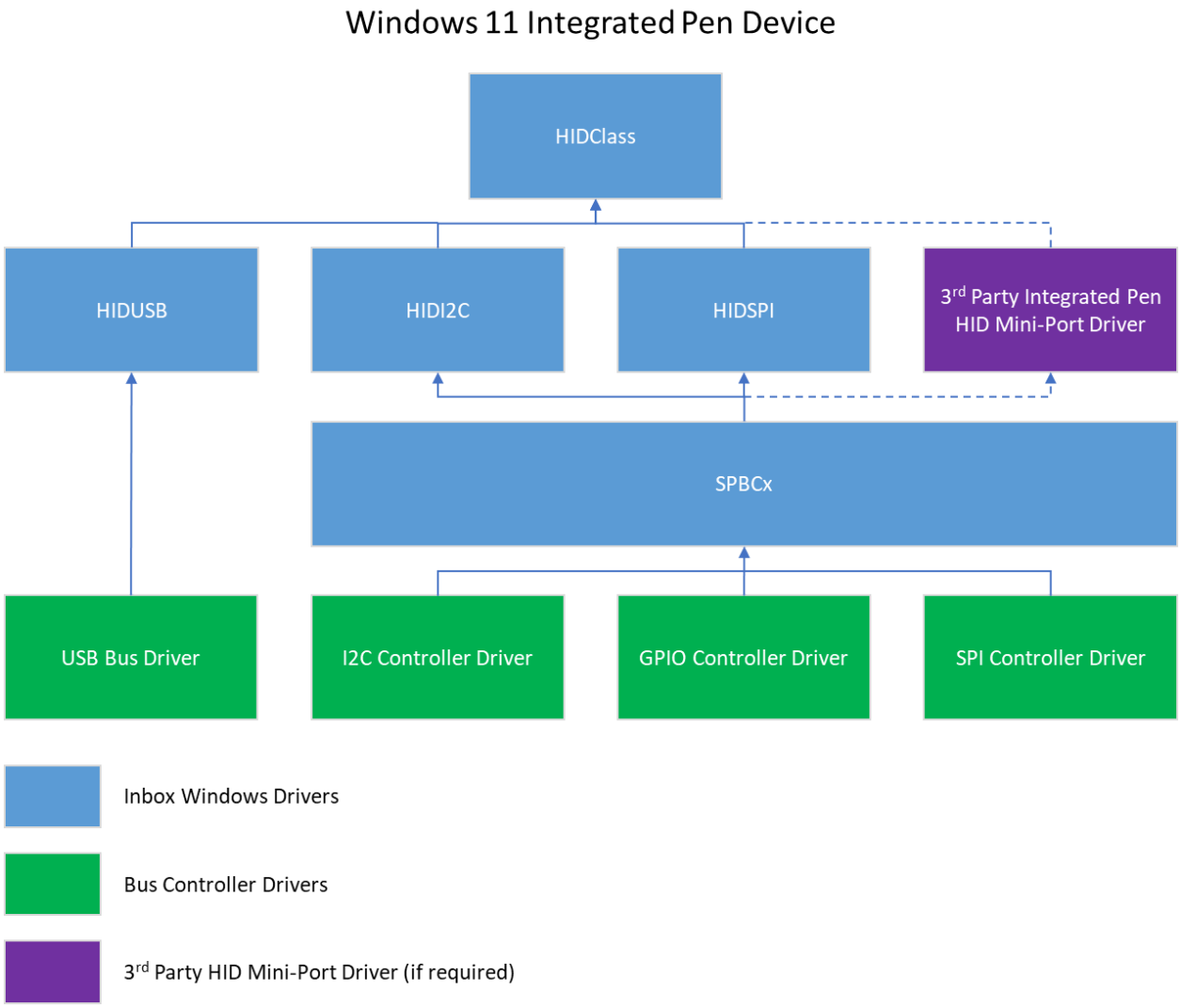
Device Bus Connectivity (pen-device-bus-connectivity)

Article • 06/21/2022

> **Note**
>
> This article contains references to the term *slave*, a term that Microsoft no longer uses. When the term is removed from the software, we'll remove it from this article.

This topic discusses bus connectivity methods for a Windows pen device.

An integrated Windows pen device can use the Microsoft-provided inbox drivers to connect to its Windows host, using either HID over USB, HID over I^2C, or (starting in Windows 11) HID over SPI. However you can use any other bus that you want, as long as you provide the required Windows-compatible, 3rd party HID mini-port driver for the pen device. The following diagram shows the Windows 11 driver stack for a Windows pen device.



For full compatibility with Windows 11 for desktop editions (Home, Pro, Enterprise, and Education), we recommend using the Microsoft-provided inbox drivers. If you decide to use a 3rd party mini-port driver, then you must add this 3rd party driver to the appropriate OEM, and System Restore images, and then make these images available for download on Windows Update.

The following sections present some examples of device configurations.

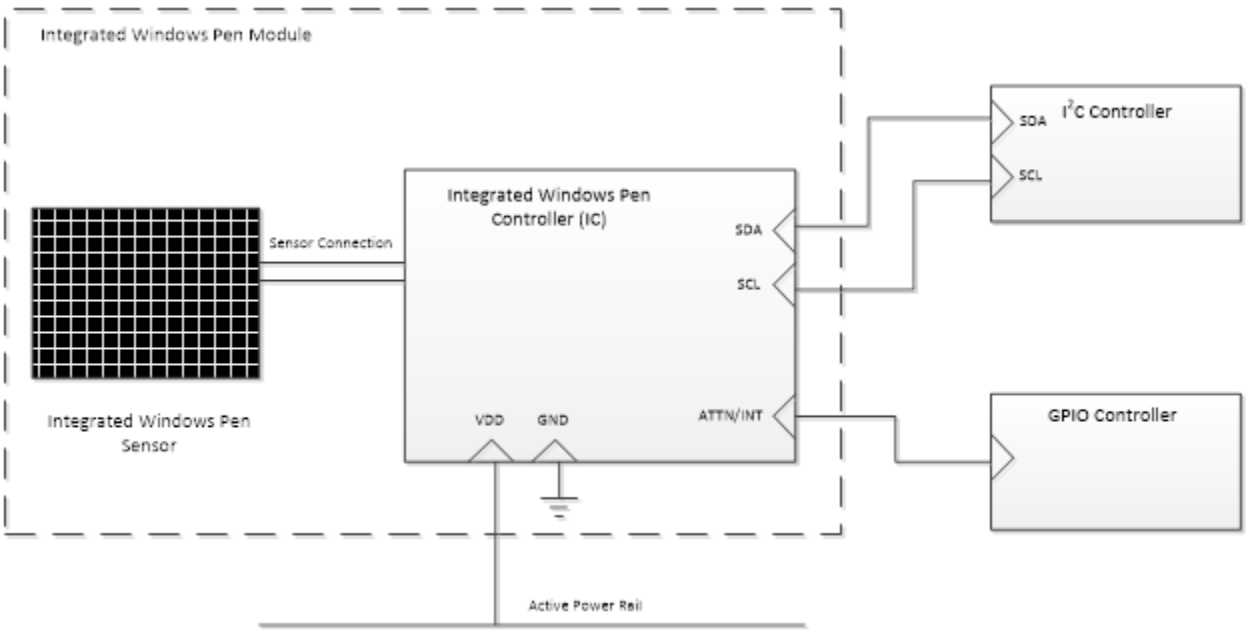
I²C Devices

An integrated Windows pen module is defined as the combination of a controller IC and a sensor.

A Windows pen module that connects to its Windows host via the I²C bus must, at a minimum, expose the following five connection pins:

- A data line (SDA)
- A clock line (SCL)
- An interrupt line
- A power supply line
- A ground connection (GND)

The following is a diagram of the connections lines between a Windows pen device and its Windows host.



When connecting to an I²C controller it is important to understand the bandwidth demands of all the components sharing that controller. A minimum I²C clock speed of 400 KHz is recommended for an integrated Windows pen. It is highly recommended

that integrated Windows pen controllers do not share the same I²C controller with components that have high bandwidth usage.

We recommend connecting the interrupt line (also referred to as an ATTN line) to an On-SoC GPIO controller, or to an IOAPIC. The GPIO or IOAPIC resource to which the interrupt line is connected, should be capable of (and configured for) waking the SoC. The wake-up capability allows the integrated Windows pen to wake the system in various scenarios.

If you decide to use the wake functionality, the power line that is connected to the integrated Windows pen device should not be shared with other devices which are not wake-capable. In order for wake scenarios to function properly, the power line that is used must be energized during connected standby/S3 conditions.

ACPI Table Entries

A Windows pen device that is connected via I²C must define an entry in the Advanced Configuration and Power Interface (ACPI) table of the host, for the device to be recognized by the host. For more information about ACPI, see Advanced Configuration and Power Interface Specification.

The ACPI table entry should specify the following information:

Entry	Description
ACPI Device Entry Name	A 4-character identification unique to the ACPI table, to reference the device. For example, “WPEN”.
ACPI Hardware ID	A 4-character + 4-number ACPI hardware ID, to reference the device. This is exposed in device manager. For example, “MSFT0001”.
Compatible ID	This should always be “PNP0C50” to indicate that the device is HID I²C compatible.
I²C Controller	Specifies an I²C controller on the Windows host. This controller is used to connect the pen to the Windows host, and makes it possible for the pen and the host to communicate. For example, “I2C3” – to indicate I²C controller #3.
I²C Slave Address	Specifies the I²C slave address for the device. The host uses this address to single out the pen device on the I²C bus for communication. For example, “0x6F”.
I²C Speed	Specifies the maximum speed supported by both the device and the I²C controller. Specifying the speed in the ACPI table ensures reliable communication. This speed should not be any lower than 400KHz (0x61A80).
GPIO Controller	The GPIO controller to which the pen's interrupt line is connected. This tells the host where to "listen" for interrupt signals. For example, “GPIO0” – to indicate GPIO controller #0.

Entry	Description
GPIO Resource/Pin	The GPIO controller pin to which pen's interrupt line is connected. The host then associates this specific GPIO pin with interrupt signals from the pen. For example, “{35}” – to indicate pin 35.
GPIO Resource Type	Defines the constraints around the GPIO resource. This entry for the ACPI table, should be set to “Exclusive” unless you want to select SoC wake. If you decide to select SoC Wake, then set this entry to “ExclusiveAndWake”.
GPIO Interrupt Assertion Type	Defines the type of triggering that the pen will provide for its interrupts. This can either be "Edge-triggered", or "Level-triggered". HID I^2C-compliant devices should use “Level-triggered" interrupts.
GPIO Interrupt Assertion Level	Defines the voltage level on the interrupt line, when the interrupt is asserted by the device. This can be specified as “ActiveLow” or “ActiveHigh”.

USB Devices

A high-speed/full-speed integrated Windows pen module that is connected via USB 2.0, should expose the necessary pins for host connectivity.

Connection to the host can take many forms and is at the discretion of the integrator.

Note that, when connecting to a USB hub, it is important to understand the bandwidth demands of all the components that share the hub. It is highly recommended that high-bandwidth devices, and integrated Windows pen controllers do not share the same USB hub, as this may result in bandwidth demands that exceed bus capability.

USB Bridge Devices (I^2C -> USB)

If you use a USB bridge to connect an integrated I^2C Windows pen to the host, the bridge should expose the integrated Windows pen as a distinct device node, with the device’s unique attributes (wVendorID, wProductID, wVersionID).

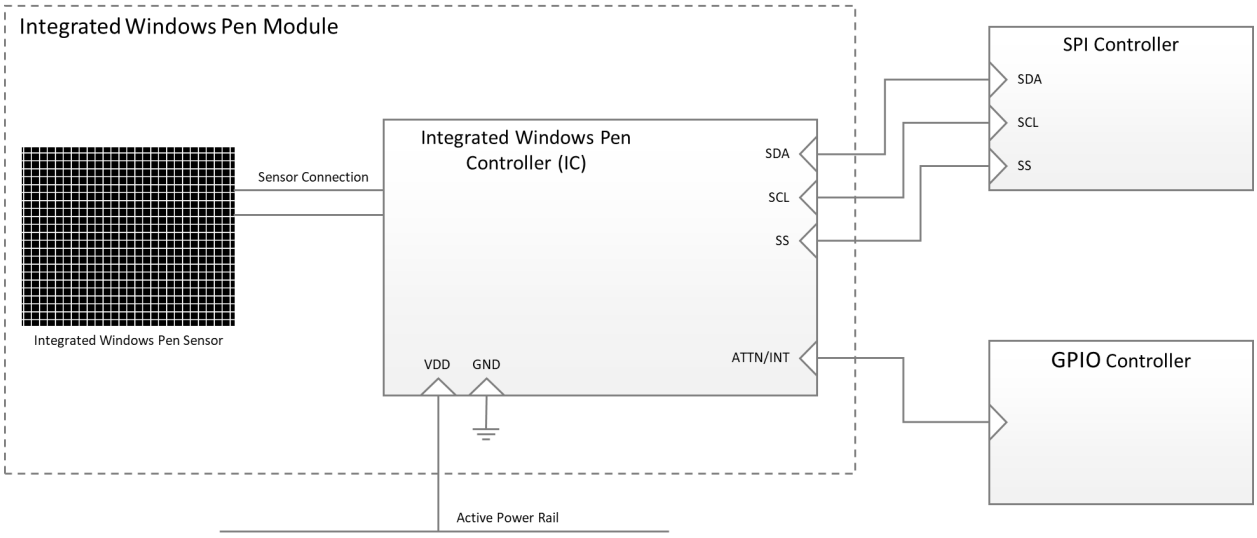
HID SPI Devices

An Windows Pen module is defined as the combination of a controller IC and a sensor.

A Windows pen module that connects to its Windows host via the SPI bus must, at a minimum, expose the following five connection pins:

- A data line (SDA)
- A clock line (SCL)
- A chip select line (SS)
- An interrupt line
- A power supply line
- A ground connection (GND)

The following is a diagram of the connections lines between a Windows Pen device and its Windows host.



ACPI Table Entries

A Windows Pen device that is connected via SPI must define an entry in the Advanced Configuration and Power Interface (ACPI) table of the host, for the device to be recognized by the host. For more information about ACPI, see Advanced Configuration and Power Interface Specification.

The ACPI table entry should specify the information as described in the Plug and Play support for HID over SPI article.

Power Management (pen-power-management)

Article • 09/30/2021

This topic provides information about pen power management, and discusses power consumption requirements.

Power Consumption

Power consumption requirements for the various power modes of an integrated Windows pen are at the discretion of the OEM, or the system builder. However, Windows provides facilities to help balance power efficiency and response latency.

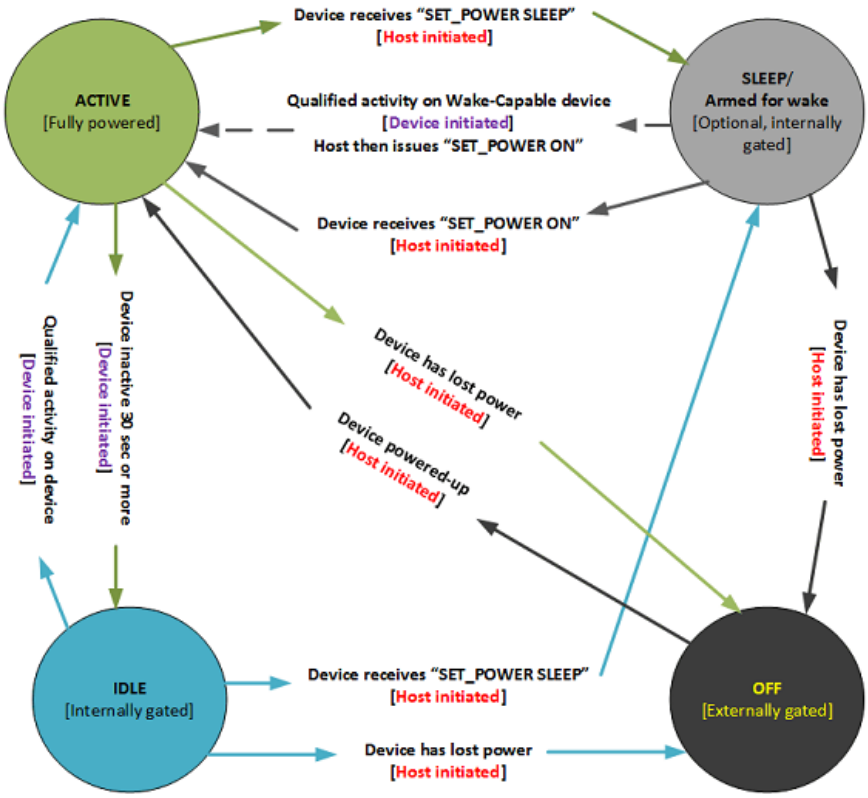
> **Note:** If the Pen device is connected via I2C, SPI, or USB or SPI, wake on pen is still not supported on a modern standby system. It will, however, work on an S3 system.

I²C Devices

Integrated Windows pen devices that are connected via I²C can implement support for up to four distinct power states:

- Active
- Idle
- Either Sleep, or Armed for Wake (both optional)
- Off

The four power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which the pen is in-range of the screen surface, or the operating mode in which there has been activity within the last 30 seconds. When power is applied to a pen device, then after device boot-up is completed, the device should be ready and in the **Active** power state.

Idle State

The **Idle** state is defined as the device operating mode in which no activity has occurred within the last 30 seconds.

A device can elect to reduce its scan rate in this mode, to reduce power consumption while still adhering to the down latency requirement for this mode. Once the device has detected activity, it should transition back to the **Active** state.

Sleep (Armed for Wake) State

The **Sleep** state is defined as the operating mode that the device enters, after the host sends a HID I^2C "SET_POWER SLEEP" command to the device.

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of asserting an interrupt to wake the system, in response to a qualified activity. An integrated Windows pen device should ensure that

interrupts are not asserted for spurious contacts, since such contacts would result in an unintended system wake. There are no down latency requirements for this mode.

The device should transition to the **Active** state after it receives a HID I^2C "SET_POWER ON" command from the host.

Off State

The **Off** state is defined as the operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed, the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

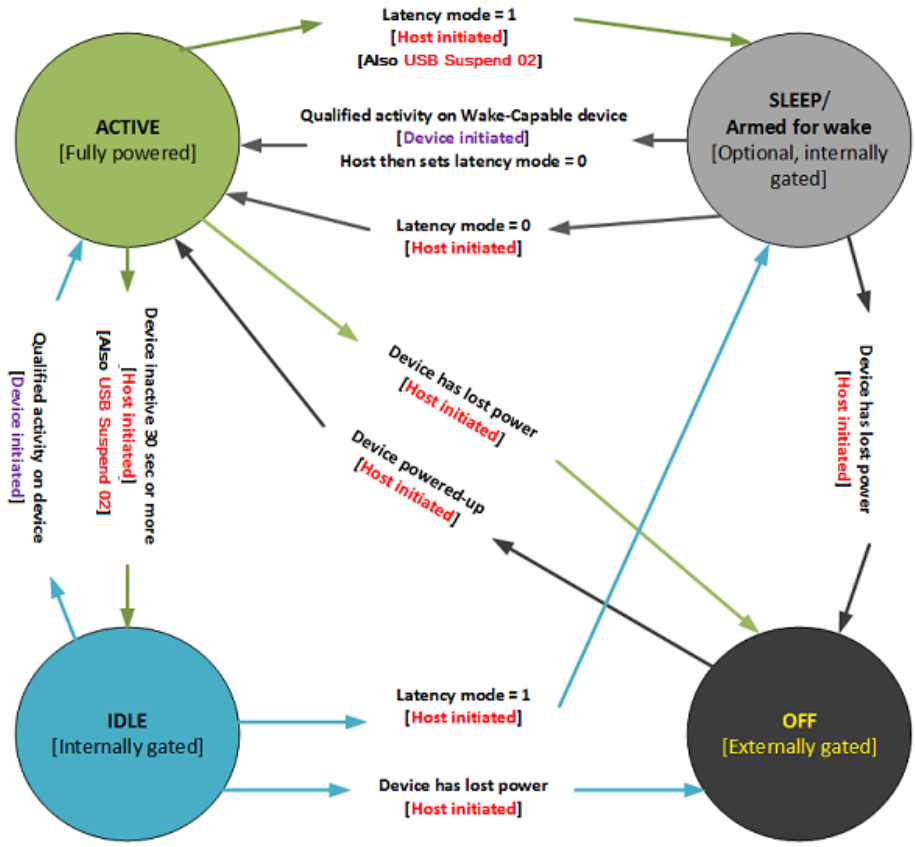
In the **Off** state a device should not consume any power.

USB Devices

Integrated Windows pen devices that connect to the host via USB, can implement support for up to four distinct power states:

- Active
- Idle
- Either Sleep, or Armed for Wake (both optional)
- Off

The four power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which the host has not suspended the device. When power is applied to a pen device, then after device boot-up is completed, the device should be ready, and in the **Active** power state.

A device should adhere to the contact down latency, and contact move latency requirements for this mode.

Idle State

The **Idle** state is defined as the device operating mode in which no contact or button activity has occurred within a host-defined period, and the device has therefore been suspended. This is referred to as "*USB selective suspend*."

All integrated Windows pen devices that are connected to their host via USB, should support selective suspend. Such devices should report this capability via a Microsoft OS descriptor.

A pen device can elect to reduce its scan rate in this mode, to reduce power consumption while still adhering to the down latency requirement for this mode.

Once the device has detected qualified activity, it should signal a remote wake. Starting from the time of detection of the activity, the device should buffer at least 100ms worth of contact reports, to ensure that little to no input is lost while the USB host controller is resuming.

Sleep (Armed for Wake) State

The **Sleep** state is defined as the device operating mode in which the host has transitioned to S3 or *Connected Standby*. This is indicated to the pen device via the latency mode feature report, with a value of "1" indicating that maximum latency is permitted. The device should exit this high latency mode upon activity detection, and also in response to "host resume."

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of signaling a remote wake (in response to qualified activity) to wake the system. An integrated Windows pen should ensure that remote wake is not signaled in response to spurious contacts, which would result in an unintended system wake. There is no contact down latency requirements for this mode.

Off State

The **Off** state is defined as the device operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed, the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

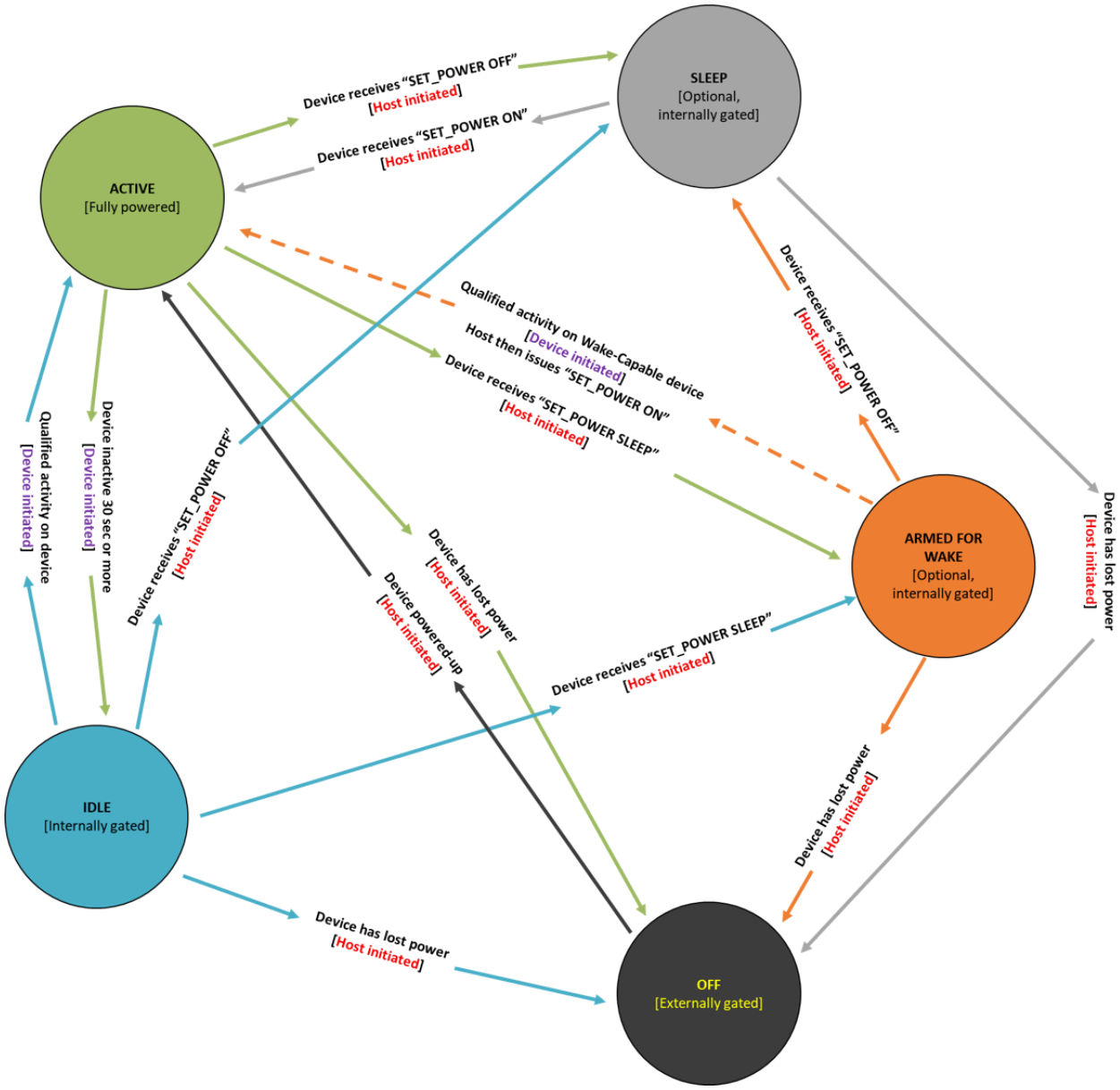
In the **Off** state a pen device should not consume any power.

SPI Devices

Windows Pen devices that are connected via SPI can implement support for up to five distinct power states:

- Active
- Idle
- Sleep (optional)
- Armed for Wake (optional)
- Off

The five power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which one or more contacts are present, or there has been activity within the last 30 seconds. When power is applied to a Pen device, then after device boot-up is completed, the device should be ready and in the **Active** power state.

A Pen device should adhere to the *contact down latency* and *contact move latency* requirements for this mode, as this is where the bulk of the compatibility requirements will be tested, and where the majority of user interactions will occur.

Idle State

The **Idle** state is defined as the device operating mode in which no activity has occurred within the last 30 seconds.

A device can elect to reduce its scan rate in this mode, to reduce power consumption while still adhering to the contact down latency requirement for this mode. Once the

device has detected activity, it should transition back to the **Active** state.

Sleep State

The **Sleep** state is defined as the operating mode that the device enters, after the host sends a HID SPI "SET_POWER OFF" command to the device.

In this state, the device should enter its lowest possible internal power state and should not be scanning for input or be armed to wake.

The device should transition to the **Active** state after it receives a HID SPI "SET_POWER ON" command from the host.

Armed for Wake State

The **Armed for Wake** state is defined as the device operating mode in which the host has transitioned to a lower power state, but is still scanning for input, so it is configured to be woken up by pen interactions.

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of signaling a remote wake (in response to qualified activity) to wake the system. A Windows Pen device should ensure that remote wake is not signaled in response to spurious contacts, which would result in an unintended system wake. There are no contact down latency requirements for this mode.

A contact occurrence that causes entry into the armed for wake state should not be reported to wake the system.

The device should transition to the **Active** state after it receives a HID SPI "SET_POWER ON" command from the host.

Off State

The **Off** state is defined as the operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed, the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

In the **Off** state a device should not consume any power.

Protocol Implementation (pen-protocol-implementation)

Article • 03/18/2021

This section provides guidelines for pen protocol implementation. Integrated Windows pen devices are expected to use the Human Interface Device (HID) protocol to communicate with the host.

A good understanding of the HID protocol is needed to be able to understand the information presented here. See the following resources for information about the HID protocol:

- Device Class Definition for Human Interface Devices (HID) Version 1.11
- HID Usage Tables Version 1.12
- HID Over I2C Protocol Specification Version 1.0

Windows includes a HID class driver and corresponding HID I^2C, HID USB and HID BTH miniport drivers. So there is no need to develop 3rd-party drivers for integrated Windows pen devices that use one of these buses. In fact, the Windows 10 Hardware Lab Kit (HLK) doesn't even allow the testing of 3rd-party drivers for Windows pen devices that connect via one of the buses mentioned previously. However, if you want to use a 3rd-party bus controller, or develop your own to use with your Windows pen device, you can do that.

If you want your pen device to connect via an alternate bus, then a 3rd-party HID mini-port driver might be required, and would be allowed according to the Windows 10 HLK requirements. You only need to report the usages described in this topic, in the firmware for your integrated Windows pen device. Windows will use your firmware and its own HID drivers to enable pen and inking capabilities for your device, and also provide Windows applications access to your device.

A sample descriptor is provided in the Sample Report Descriptors topic.

In this section

Topic	Description
Required HID Descriptors	This topic presents the required HID descriptors (and device attributes) for a Windows pen device in Windows 10 and later operating systems.

Topic	Description
Required HID Top-Level Collections	This topic discusses the required HID top-level collections that are used for pen reporting in Windows 10 and later operating systems.
Windows Pen States	This topic discusses the pen states for a Windows pen device in Windows 10 and later operating systems.
Sample Report Descriptors	This topic presents a sample report descriptor and some sample descriptor excerpts for a Windows pen device.

Required HID Descriptors (required-hid-descriptors)

Article • 03/18/2021

This topic presents the required HID descriptors (and device attributes) for a Windows pen device in Windows 10 and later operating systems.

Required USB HID Descriptor

The following table shows the required USB HID descriptor. For more information, see **section 6.2.1** in Device Class Definition for Human Interface Devices (HID) Version 1.11.

Member	Size in bytes	Description
bLength	1	Size of the descriptor
bDescriptorType	1	Type of descriptor
bcdHID	2	HID version number
bCountryCode	1	Country code
bNumDescriptors	1	Number of descriptors
bDescriptorType	1	Descriptor type
bDescriptorLength	2	Length of the descriptor

Required I²C HID Descriptor

The following table shows the required I²C HID descriptor.

Member	Size in bytes	Description
wHIDDescLength	2	The length of the complete HID descriptor (in Bytes).
bcdVersion	2	The version number, in binary coded decimal (BCD) format.
wReportDescLength	2	The length of the Report descriptor (in Bytes).
wReportDescRegister	2	The register index containing the Report descriptor.

Member	Size in bytes	Description
wInputRegister	2	The register number to read the input report (in unsigned Bytes).
wMaxInputLength	2	The length of the largest input report to be read from the input register.
wOutputRegister	2	The register number to send the output (in unsigned Bytes).
wMaxOutputLength	2	The length of the largest output report to be sent.
wCommandRegister	2	The register number to send command requests (in unsigned Bytes).
wDataRegister	2	The register number to exchange data with command requests (in unsigned Bytes).
wVendorID	2	USB-IF assigned Vendor ID.
wDeviceID	2	Device ID.
wVersionID	2	Firmware version number.

Required Device Attributes

The following HID properties must be provided in the device attributes. The reporting of these device attributes is bus-specific. Consult the HID-specific guidance for your choice of bus.

Member	Description	USB	I^2C
wVendorID	Vendor ID	idVendor in USB Device Descriptor	wVendorID in I^2C HID Descriptor (see the preceding table).
wProduct	Product ID	idProduct in USB Device Descriptor	wDeviceID in I^2C HID Descriptor (see the preceding table).
wVersionID	Firmware version number	bcdDevice in USB Device Descriptor	wVersionID I^2C HID Descriptor (see the preceding table).

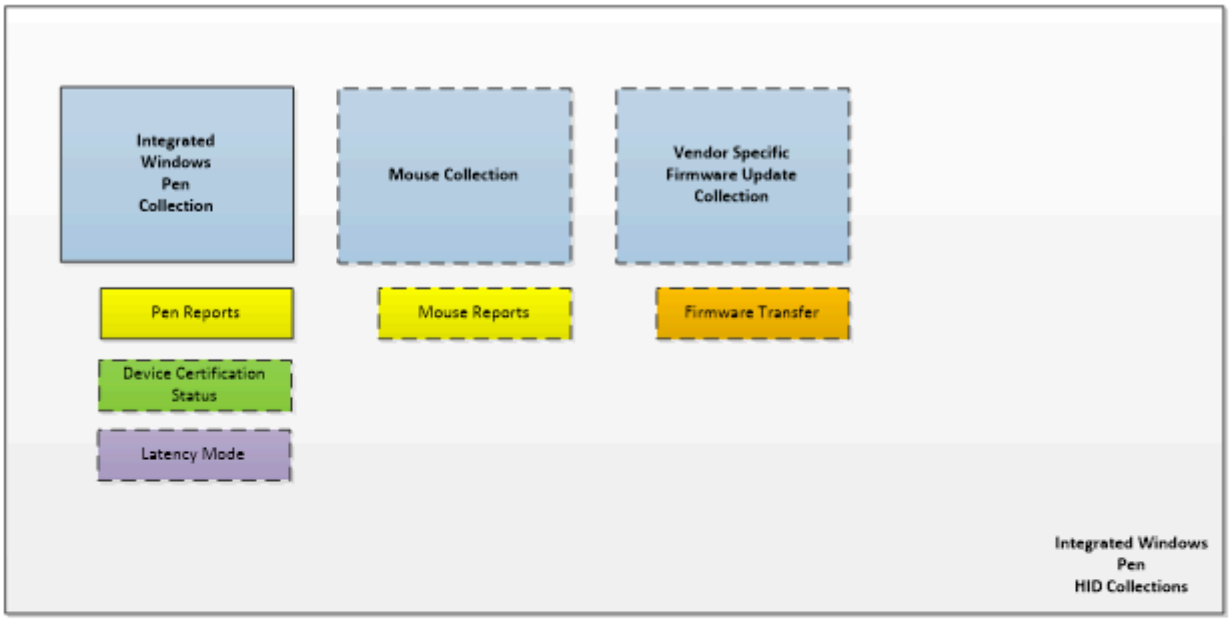
Required HID Top-Level Collections (required-hid-top-level-collections)

Article • 04/17/2024

This topic discusses the required HID top-level collections that are used for pen reporting in Windows 10 and later operating systems.

An integrated Windows pen device should expose, at a minimum, the single mandatory top-level collection for pen reporting. An optional (but recommended) collection for firmware updates can also be implemented. And on operating systems earlier than Windows 10, an optional (dummy) mouse collection for pen cursor support can also be implemented.

The following diagram shows the HID collections for an integrated Windows pen device.



Mouse Collection (Dummy)

An integrated Windows pen device can use the HID protocol to provide a top-level collection that appears as a generic desktop/mouse (Page **0x01**, Usage **0x02**).

The (dummy) mouse collection of an integrated Windows Pen device serves the purpose of providing the host with a HID-compliant mouse device, for the purposes of displaying a cursor on operating systems earlier than Windows 10. No actual mouse reporting is required from this collection; only the presence of the mouse collection in the descriptor. If you don't have to support operating systems earlier than Windows 10, then this collection can be omitted.

Integrated Windows Pen Collection

An integrated Windows pen device should use the HID protocol on a Windows 10 system in such a way that the device provides a top-level collection that appears as a digitizer/integrated pen (Page **0x0D**, Usage **0x02**).

The integrated Windows pen collection serves the purpose of providing rich contact, pressure, erase and button reporting to the host. The collection can optionally support a feature report to obtain the device's certification status. The mandatory input report is specified in detail the *Integrated Windows Pen Input Reports* section of this topic. An optional (but highly recommended) feature report can be implemented to obtain latency mode hints from the host. This optional feature can help to achieve minimal power consumption on USB devices that are armed for wake in sleep mode.

Device Certification Status Feature Report

If you implement the device certification status feature report, the host of the integrated Windows Pen device can use it to retrieve the device's 256-byte blob.

The 256-bytes should be specified via a vendor-specific HID usage, in a vendor-defined usage page (Page **0xFF**, Usage **0xC5**) in the device certification status feature report.

Before a device receives a 256-byte blob that indicates its certification status, it should implement a default blob as follows:

0xfc, 0x28, 0xfe, 0x84, 0x40, 0xcb, 0x9a, 0x87, 0x0d, 0xbe, 0x57, 0x3c, 0xb6, 0x70, 0x09, 0x88, 0x07, 0x97, 0x2d, 0x2b, 0xe3, 0x38, 0x34, 0xb6, 0x6c, 0xed, 0xb0, 0xf7, 0xe5, 0x9c, 0xf6, 0xc2, 0x2e, 0x84, 0x1b, 0xe8, 0xb4, 0x51, 0x78, 0x43, 0x1f, 0x28, 0x4b, 0x7c, 0x2d, 0x53, 0xaf, 0xfc, 0x47, 0x70, 0x1b, 0x59, 0x6f, 0x74, 0x43, 0xc4, 0xf3, 0x47, 0x18, 0x53, 0x1a, 0xa2, 0xa1, 0x71, 0xc7, 0x95, 0x0e, 0x31, 0x55, 0x21, 0xd3, 0xb5, 0x1e, 0xe9, 0x0c, 0xba, 0xec, 0xb8, 0x89, 0x19, 0x3e, 0xb3, 0xaf, 0x75, 0x81, 0x9d, 0x53, 0xb9, 0x41, 0x57, 0xf4, 0x6d, 0x39, 0x25, 0x29, 0x7c, 0x87, 0xd9, 0xb4, 0x98, 0x45, 0x7d, 0xa7, 0x26, 0x9c, 0x65, 0x3b, 0x85, 0x68, 0x89, 0xd7, 0x3b, 0xbd, 0xff, 0x14, 0x67, 0xf2, 0x2b, 0xf0, 0x2a, 0x41, 0x54, 0xf0, 0xfd, 0x2c, 0x66, 0x7c, 0xf8, 0xc0, 0x8f, 0x33, 0x13, 0x03, 0xf1, 0xd3, 0xc1, 0x0b, 0x89, 0xd9, 0x1b, 0x62, 0xcd, 0x51, 0xb7, 0x80, 0xb8, 0xaf, 0x3a, 0x10, 0xc1,

0x8a, 0x5b, 0xe8, 0x8a, 0x56, 0xf0, 0x8c, 0xaa, 0xfa, 0x35, 0xe9, 0x42, 0xc4, 0xd8, 0x55, 0xc3, 0x38, 0xcc, 0x2b, 0x53, 0x5c, 0x69, 0x52, 0xd5, 0xc8, 0x73, 0x02, 0x38, 0x7c, 0x73, 0xb6, 0x41, 0xe7, 0xff, 0x05, 0xd8, 0x2b, 0x79, 0x9a, 0xe2, 0x34, 0x60, 0x8f, 0xa3, 0x32, 0x1f, 0x09, 0x78, 0x62, 0xbc, 0x80, 0xe3, 0x0f, 0xbd, 0x65, 0x20, 0x08, 0x13, 0xc1, 0xe2, 0xee, 0x53, 0x2d, 0x86, 0x7e, 0xa7, 0x5a, 0xc5, 0xd3, 0x7d, 0x98, 0xbe, 0x31, 0x48, 0x1f, 0xfb, 0xda, 0xaf, 0xa2, 0xa8, 0x6a, 0x89, 0xd6, 0xbf, 0xf2, 0xd3, 0x32, 0x2a, 0x9a, 0xe4, 0xcf, 0x17, 0xb7, 0xb8, 0xf4, 0xe1, 0x33, 0x08, 0x24, 0x8b, 0xc4, 0x43, 0xa5, 0xe5, 0x24, 0xc2

The host can request the device certification status feature report of an integrated Windows pen device at any time, after reading the report descriptor.

Latency Mode Feature Report

> **Note**
>
> Windows 10, version 1511 and earlier operating systems do not currently issue this HID feature report to Pen or Touchscreen devices that happen to provide support for it. However, adding support for the *Latency Mode Feature Report* to a device, will make sure that the device is ready for the future date when Windows supports it on Pen and Touchscreen devices.

The latency mode feature report is sent by the host to an integrated Windows pen device, to indicate when high latency is desirable for power savings and conversely, when normal latency is desired for operation. In the case of a USB-connected integrated Windows pen device, this allows the device to differentiate between being suspended for inactivity (runtime IDLE) and being suspended because the system is entering S3 or Connected Standby.

The latency mode should be indicated via the value for the latency mode usage (Page **0x0D**, Usage **0x60**) in the latency mode feature report.

Expand table

Latency mode value	Latency mode
0	Normal
1	High

Integrated Windows Pen Input Reports

The usages in the following table are used by the host to extract pen data from an input report, via the integrated Windows pen collection. The table shows all the mandatory usages and the Windows-supported optional usages for reporting about the pen.

Expand table

Member	Description	Page	ID	Mandatory/Optional	Required for HLK
X	X coordinate of contact position	0x01	0x30	Mandatory	Yes
Y	Y coordinate of contact position	0x01	0x31	Mandatory	Yes
Tip	Set if the pen is on the surface of the digitizer	0x0D	0x42	Mandatory	Yes
In-Range	Indicates whether the pen is in-range of the digitizer	0x0D	0x32	Mandatory	Yes
Barrel Button	State of button located on the side of the pen	0x0D	0x44	Mandatory	Yes
Invert	Indicates that the pen's orientation implies an intention to erase	0x0D	0x3C	Optional	Yes
Eraser	Indicates that the pen is erasing	0x0D	0x45	Optional	Yes
Tip Pressure	Pressure exerted on the pen tip	0x0D	0x30	Optional	Yes
Scan Time	Relative scan time per frame	0x0D	0x56	Optional	No
X-Tilt	The angle of tilt of the pen along the x-axis	0x0D	0x3D	Optional	No
Y-Tilt	The angle of tilt of the pen along the y-axis	0x0D	0x3E	Optional	No
Twist	Clockwise rotation of the pen	0x0D	0x41	Optional	No
Transducer Serial Number	Unique persistent identifier for the transducer	0x0D	0x5B	Optional	No

Member	Description	Page	ID	Mandatory/Optional	Required for HLK
Transducer Serial Number - Part 2	Used for extending the transducer serial number by an additional 32 bits	0x0D	0x6E	Optional	No
Transducer Vendor ID	The assigned identifier for the vendor of the transducer	0x0D	0x91	Optional	No

The following table shows the mandatory report-level usages for integrated Windows pen input reports.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Report ID	Integrated Windows Pen report ID	0x0D	0x02	Mandatory

Mandatory and Windows-supported optional usages

Any device that does not report all mandatory usages at either the contact or report level, will be non-functional as an integrated Windows pen device. Mandatory usages are strictly enforced by the Windows host. Where a logical maximum value has not been set, the value can be optimized to reduce descriptor size.

The following sections provide more details about the mandatory and the Windows-supported optional usages that were presented in an earlier table.

X, Y

X and *Y* report the coordinates of the integrated pen. The following global items should be specified for both the *X* and *Y* usages:

- Physical minimum & Physical maximum
- Unit & Unit exponent
- Logical minimum & Logical maximum (ensuring an input resolution >= 150DPI)

  Note The entire logical coordinate range should be reportable across both the *X* and *Y* axis.

Pen Tip

Used to indicate when the pen tip is on the surface, or has left the surface of the digitizer. This is indicated by a main item with a report size of 1 bit. When delivering an input report, the bit should be set when the pen is on the digitizer surface, and cleared when the pen has left the surface.

When a pen is being reported for the first time in a report stream with the tip switch clear, the X/Y location being reported should be the same as the last position reported with the tip switch set.

In-Range

Used to indicate when the pen tip (or tail-end eraser) is in-range of the digitizer. This is indicated by a main item with a report size of 1 bit. When delivering an input report, the bit should be set when the pen tip (or tail-end eraser) is on the digitizer surface or within the detectable range of the digitizer, and cleared when the pen is no longer detectable. It should be noted that In-Range should only be reported when the pen's X/Y location can be reported reliably, and within the parameters set forth by the Pen HLK requirements for Windows 10.

When a pen is being reported with the in-range switch clear, the X/Y location being reported should be the same as the last position reported when the in-range switch was set.

Barrel Button

Used to indicate when the button on the side of the pen is depressed. This is indicated by a main item with a report size of 1 bit. When delivering an input report, the bit should be set when this button is depressed, and cleared when the button is released.

Invert

Used to indicate when the pen's orientation implies an intention to erase. This is indicated by a main item with a report size of 1 bit. Here are some input reporting guidelines for the different eraser implementations:

- Tail-end eraser implementations

  When delivering an input report, the bit should be set whenever the pen is inverted, and cleared when the pen is not inverted.

- Eraser button implementations

  When delivering an input report, the bit should be set whenever the eraser button is depressed, and the pen is in-range of the digitizer and cleared otherwise.

Eraser

Used to indicate when the pen is erasing. This is indicated by a main item with a report size of 1 bit. Here are some input reporting guidelines for the different eraser implementations:

- Tail-end eraser implementations

  When delivering an input report, the bit should be set whenever the pen is inverted and in-contact with the screen, and cleared otherwise.

- Eraser button implementations

  When delivering an input report, the bit should be set whenever the eraser button is depressed and the pen tip is in-contact with the screen, and cleared otherwise.

  To avoid accidentally activating or cancelling the erase functionality in this implementation, it is highly recommended that once the pen tip is in contact with the screen, depressing or releasing the erase button should have no impact on the reporting of the eraser bit.

When a pen is being reported for the first time in a report stream with the erase switch clear, the X/Y location being reported should be the same as the last position reported when the erase switch was set.

Scan Time

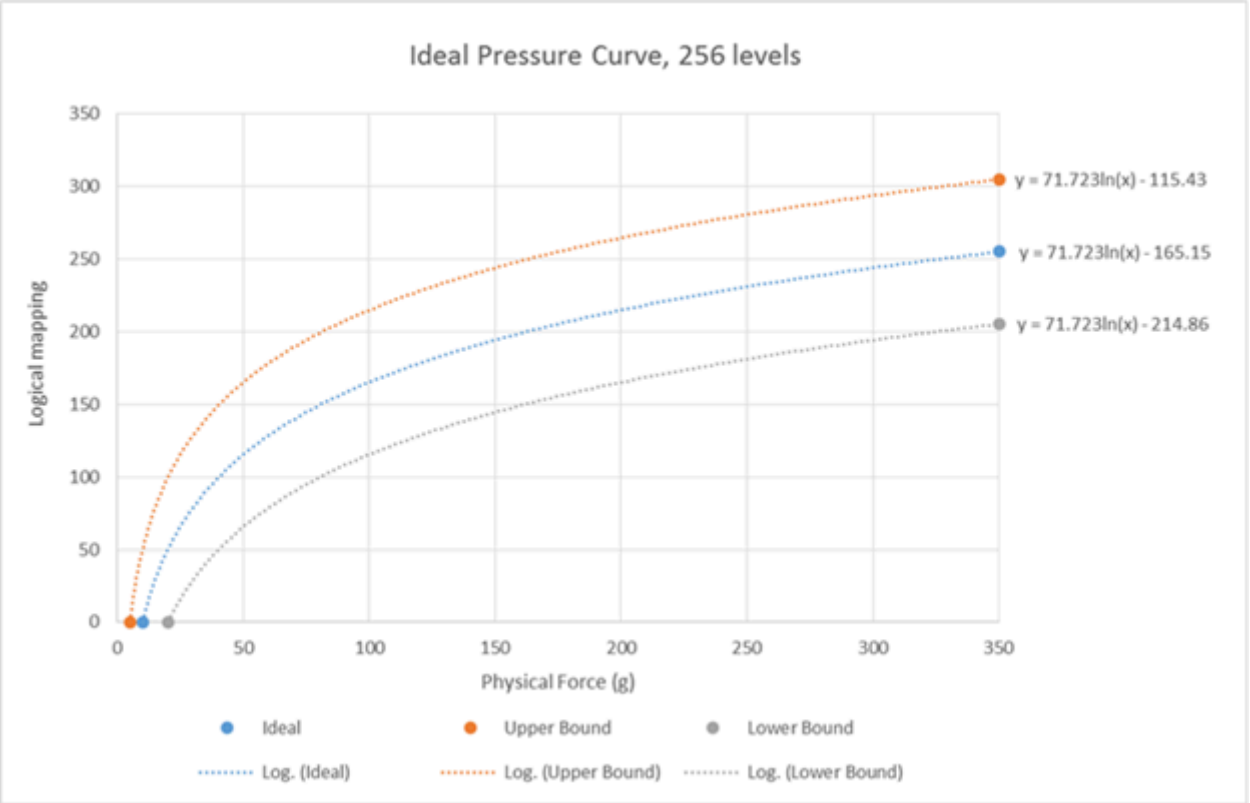
Scan Time reports relative digitizer time in 100µs units. When a device starts reporting data subsequent to a period of inactivity, the Scan Time represents the time span from the first frame that was reported till the time of the report. The first scan time received is treated as a base time for subsequent reported times. The elapsed time between reported scan times should reflect the scanning frequency of the digitizer. It is important to note that unlike other usages, the host does not allow any flexibility for the unit for the scan time usage, if it is implemented. It must be in 100µs units. The value is expected to roll over, as only 2 bytes are allocated to the counter.

Tip Pressure

Used to indicate the pressure that is being applied to the tip of the pen. This is indicated by a main item with a report size of at least 8 bits, to accommodate a minimum of 256 distinct levels of pressure. To report additional granularity, you must use a larger report size.

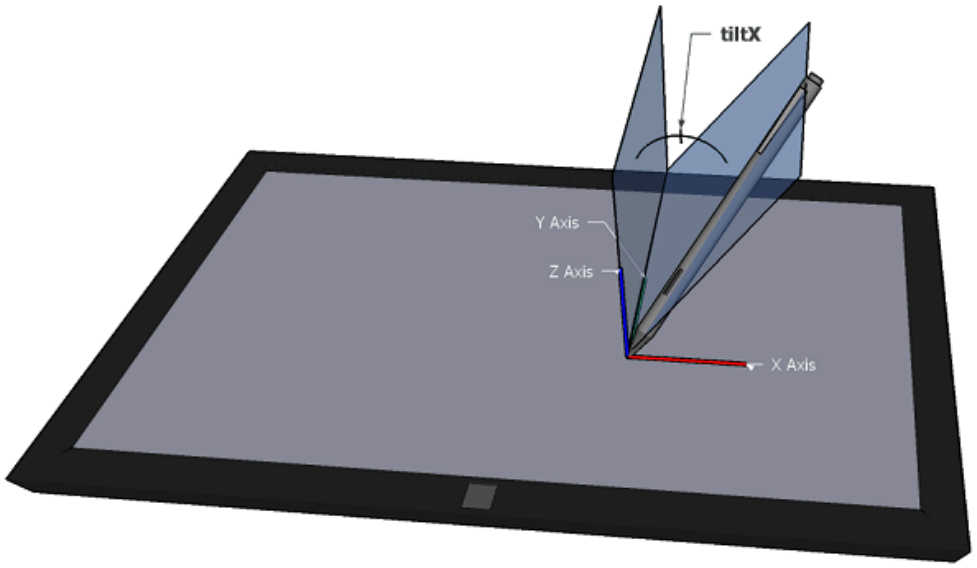
The logical values of reported pressure should adhere as closely as possible to the ideal logarithmic curve shown in the following diagram. The pressure tests provided by the Pen HLK test suite for Windows 10, are designed to ensure that the reported pressure curve falls within the upper and lower bounds indicated in the diagram. While the ideal curve in the diagram, and its associated equations, are outlined for a device reporting 256 levels, they can be scaled to accommodate any number of discrete levels.

While the eraser switch is active on pens with tail end erasers without pressure sensing capabilities, the pen tip pressure should report 127 (50% of max pressure). This value was chosen to support pens and apps that may consume real tail-end pen pressure while the eraser switch is active.



X-Tilt

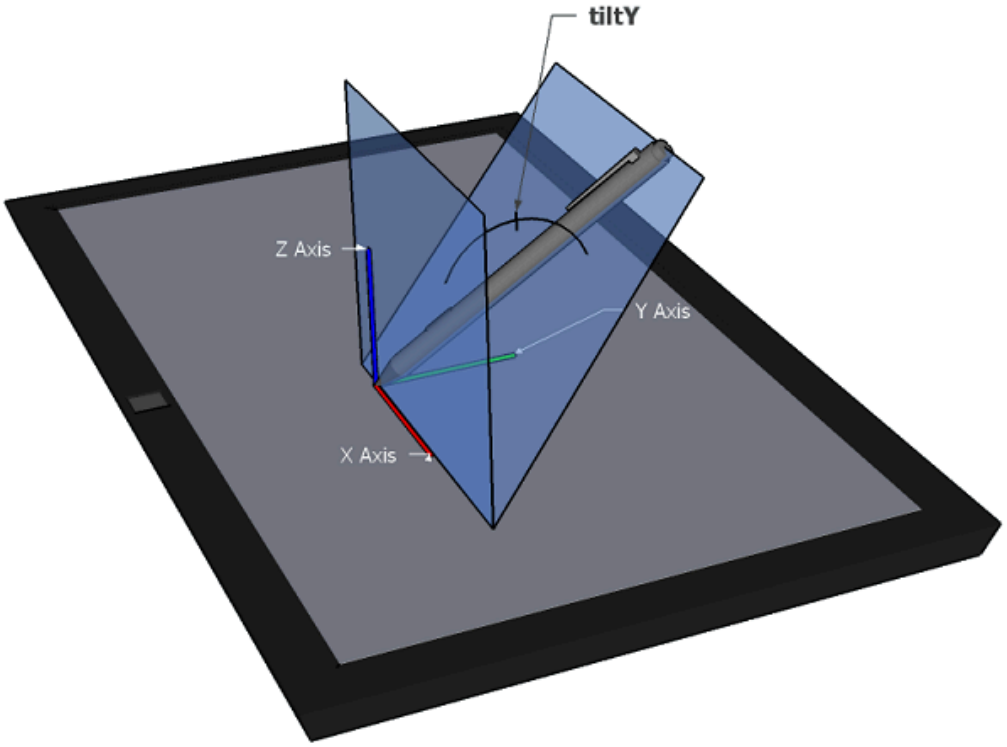
X-Tilt represents the plane angle between the Y-Z plane, the plane containing the pen transducer axis, and the Y axis.



The physical range and logical range must be specified. The physical range may be a maximum of -90 to 90 deg, however the physical minimum/maximum reported in the descriptor should correspond to the actual minimum/maximum achievable X-Tilt of the pen, in the correct unit (0x14, Degrees). The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians may also be used for the physical range. X-Tilt is positive to the right of the user in the digitizer's native orientation.

Y-Tilt

Y-Tilt represents the plane angle between the X-Z plane, the plane containing the pen transducer axis, and the Y axis.



The physical range and logical range must be specified. The physical range may be a maximum of -90 to 90 deg, however the physical minimum/maximum reported in the descriptor should correspond to the actual minimum/maximum achievable Y-Tilt of the pen in the correct unit (0x14, Degrees). The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians may also be used for the physical range. Y-Tilt is positive towards the user in the digitizer's native orientation.

Twist

Twist specifies the clockwise rotation of the pen around its own major axis.

The physical range and logical range must be specified. The physical range must be 0 to 360. The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians can also be used for the physical range. In this case, the logical range must be large enough to report values that are accurate to at least four decimal places.

Transducer Serial Number

The Transducer Serial Number is a unique persistent identifier for the transducer used in the pen accessory communicating with the pen digitizer. This is required to be 32-bits and is defined by the vendor or entity identified by the transducer vendor ID. In circumstances where the Transducer Serial Number is unknown to the digitizer either because the pen accessory does not support transmitting this value or that the transmission has not been received in full, the digitizer shall report 0 to the host. Null position is not supported by the host.

Transducer Serial Number – Part 2

Transducer Serial Number – Part 2 allows for an additional 32-bits to be specified as part of the unique persistent identifier for the transducer used in a pen accessory. In circumstances where the Transducer Serial Number – Part 2 is unknown to the digitizer either because the pen accessory does not support transmitting this value or that the transmission has not been received in full, the digitizer shall report 0 to the host. Null position is not supported by the host.

Transducer Vendor ID

The Transducer Vendor ID is a field for communicating the manufacturer of the transducer used in the pen accessory communicating with the pen digitizer. This is

required to be a 2-byte USB-IF assigned vendor ID either of the manufacturer or that of the IHV/OEM authorizing the use of their USB-IF vendor ID for this purpose.

Windows Pen States

Article • 01/18/2019

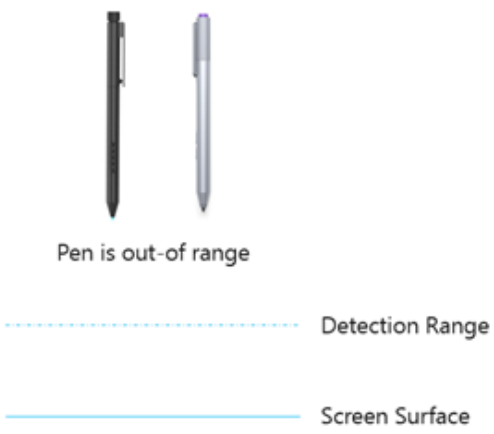
This topic discusses the pen states for a Windows pen device in Windows 10 and later operating systems.

The Main States

It is important to understand the various physical positions, or orientations in which the pen might be, the scenarios associated with these positions and how the transitions from position to position should be reported.

Out of Range

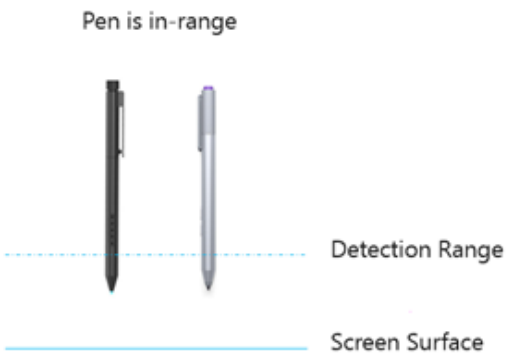
This is the simplest scenario for a Windows pen, and it occurs when the user is holding the pen out of the detection range of the digitizer.



In this state (shown in the preceding diagram), no input reports are delivered to the operating system.

In Range

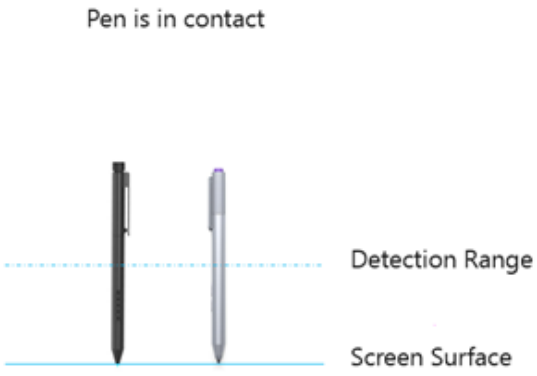
This is a common scenario for a Windows pen, and it occurs when the user is holding the pen within the detection range of the digitizer.



In this state (shown in the preceding diagram), input reports are continuously delivered to the operating system with the pen's location and the in-range switch SET. The operating system and applications can display a cursor or other feedback in response to the pen being in range.

In Contact

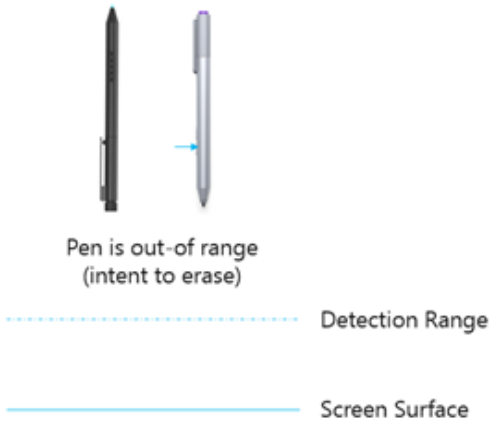
This is the most common scenario for a Windows pen, and it occurs when the user is pressing the pen against the screen surface.



In this state (shown in the preceding diagram), input reports are continuously delivered to the operating system with the pen's location, the in-range switch SET, tip switch SET, and the corresponding tip pressure. The operating system and applications can lay ink, or perform other actions in response to the pen being in contact with the screen surface.

Out of Range (Intent to Erase)

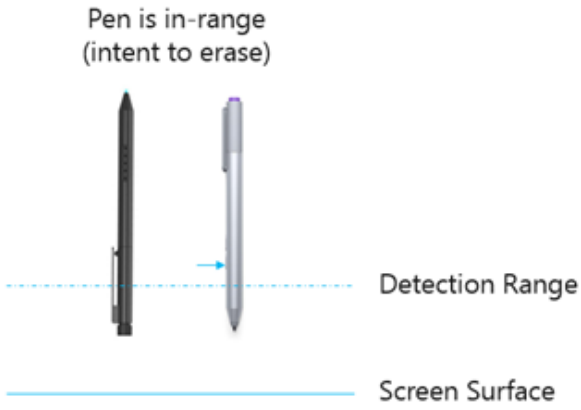
In this Windows pen scenario, the user has activated the erase capability of the pen, either by inverting it or by pressing (and holding) the erase button while the pen is out of the detection range of the digitizer. The erase capability of the pen is also referred to as the erase *affordance*.



In this state (shown in the preceding diagram), no input reports are delivered to the operating system.

In Range (Intent to Erase)

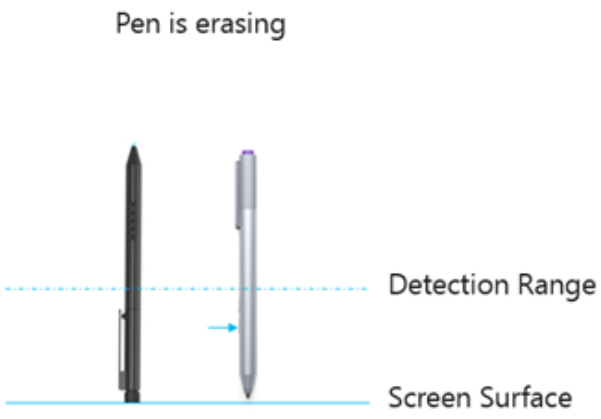
This is a common scenario for a Windows pen, and it occurs when the user is holding the pen within the detection range of the digitizer with the erase affordance activated, either by inverting the pen or by pressing (and holding) the erase button.



In this state (shown in the preceding diagram), input reports are continuously delivered to the operating system with the pen's location, the in-range switch SET and the invert switch SET. The operating system and applications can display a cursor or other feedback in response to the pen being within the detection range, with an intent to erase.

Erasing

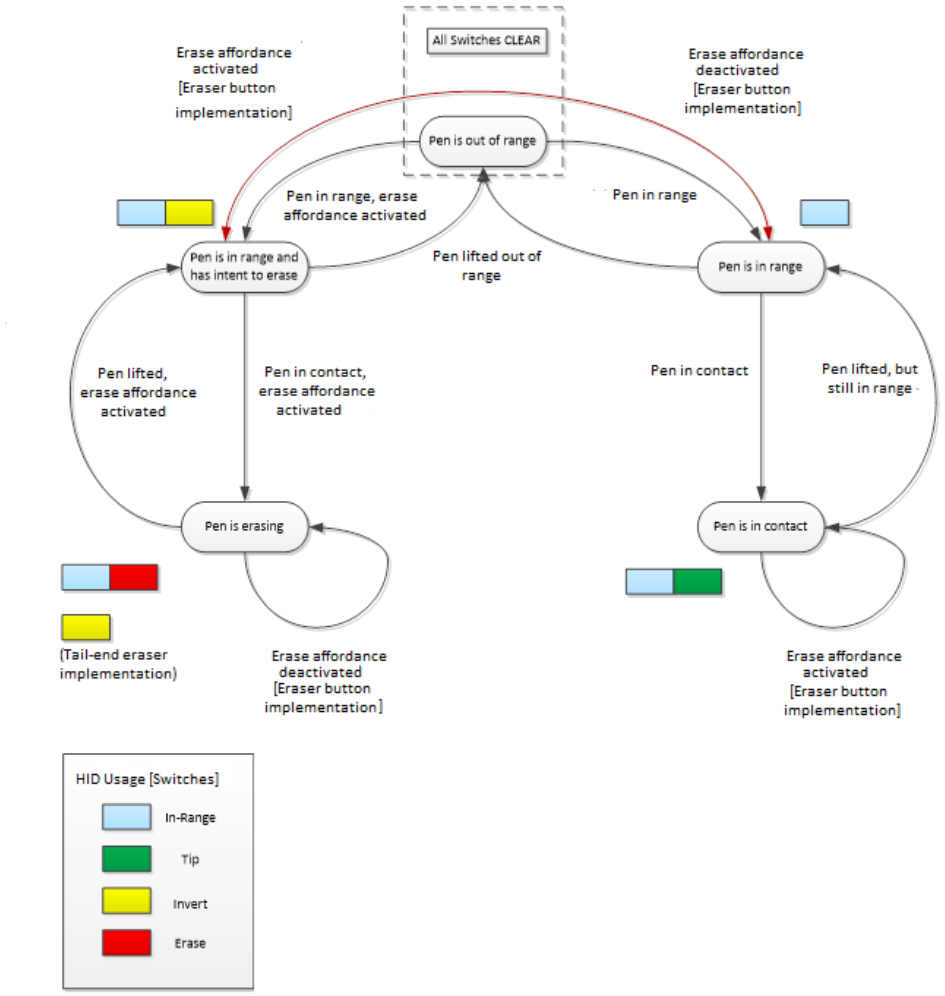
This is a common scenario for a Windows pen, and it occurs when the user is pressing the pen against the screen surface, with the erase affordance activated either by inverting the pen or by pressing (and holding) the erase button.



In this state (shown in the preceding diagram), input reports are continuously delivered to the operating system with the pen's location, the in-range switch SET, and the eraser switch SET. For pens that use a tail-end erase implementation, the invert switch should also be SET in this state. The operating system and applications can erase ink or perform other actions related to erasing.

Windows Pen State Transitions

The following diagram shows the set of valid transitions for a Windows pen (irrespective of eraser implementation). A single state is defined for out-of-range (regardless of whether the erase affordance is activated), given that the digitizer will not be sending reports to the operating system in this state.


All Switches CLEAR
Erase affordance activated [Eraser button implementation]
Erase affordance deactivated [Eraser button implementation]
Pen is out of range
Pen in range, erase affordance activated
Pen in range
Pen is in range and has intent to erase
Pen lifted out of range
Pen is in range
Pen lifted, erase affordance activated
Pen in contact, erase affordance activated
Pen in contact
Pen lifted, but still in range
Pen is erasing
Pen is in contact
(Tail-end eraser implementation)
Erase affordance deactivated [Eraser button implementation]
Erase affordance activated [Eraser button implementation]
HID Usage [Switches]
In-Range
Tip
Invert
Erase

Non-Erase Transitions

If the erase affordance is not activated (regardless of eraser implementation) when the pen enters the detection range of the digitizer, then the pen state transition that occurs, is shown in the diagram by the arc that goes from "Pen is out of range" to "Pen is in range." If the pen continues towards the screen surface and makes contact, with the erase affordance still not activated, then the diagram shows the pen state transition as going from "Pen is in range" to "Pen is in contact."

When the pen is lifted from the screen surface, and is within the detection range of the digitizer, then the arc that shows this state transition, goes from "Pen is in contact" to "Pen is in range." As part of this transition, a single input report should be delivered to the operating system, with the tip switch cleared and with the last location where the pen was in contact with the screen. As the pen continues to be lifted beyond the detection range of the digitizer, then the arc that shows this state transition, goes from "Pen is in range" to "Pen is out of range." As part of this transition, a single final input report should be delivered to the operating system with all switches cleared and with the last location where the pen was in range.

Eraser Transitions

If the erase affordance is activated (regardless of eraser implementation) when the pen enters the detection range of the digitizer, then the arc that shows this state transition, goes from "Pen is out of range" to "Pen is in range with intent to erase." If the pen continues towards the screen surface and makes contact, with the erase affordance still activated, then the arc that shows this state transition, goes from "Pen is in range with intent to erase" to "Pen is erasing."

When the pen is lifted from the screen surface and is within the detection range of the digitizer, then the arc that shows this state transition, goes from "Pen is erasing" to "Pen is in range with intent to erase." As part of this transition a single input report should be delivered to the operating system, with the eraser switch cleared and with the last location where the pen was in contact with the screen. As the pen continues to be lifted beyond the detection range of the digitizer, then the arc that shows this state transition, goes from "Pen is in range with intent to erase" to "Pen is out of range." As part of this transition, a single final input report should be delivered to the operating system, with all switches cleared and with the last location where the pen was in range.

Special Notes for Eraser Button Implementations

Unlike tail-end eraser implementations, button-based implementations can physically allow the user to activate/deactivate the erase affordance without the pen transitioning through the "out of range" state. However, this is not supported by the underlying protocol.

It is highly recommended that, while a pen with an eraser button is in contact with the screen, the eraser switch state be persisted until the pen is lifted, regardless of whether the button is pressed or released. Accidental eraser button presses during the "Pen is in contact" state, and accidental eraser button releases during the "Pen is erasing" state, are common occurrences for users, and the resulting transitions via the "Pen is out of range" state can result in a very jarring user experience.

While the pen is within detection range of the digitizer, but not in contact with the screen, activation/deactivation of the erase affordance should be honored. However, direct transitions between the "Pen is in range" and the "Pen is in range with intent to erase" states are not supported, and in this scenario, the pen states must always transition via "Pen is out of range."

For example, if the erase button is pressed while the pen is within detection range of the digitizer, but not in contact with the screen, a single input report should be delivered with all switches cleared and with the last location where the pen was in-range, followed by continuous reports where the invert switch is SET and the in-range switch is SET.

In a reverse example, if the erase button is released while the pen is within detection range of the digitizer, but not in contact with the screen, a single input report should be delivered with all switches cleared and with the last location where the pen was in-range, with the invert switch SET. This should then be followed by continuous reports where the invert switch is clear and the in-range switch is SET.

Firmware Update Collection (Optional)

An integrated Windows pen device can use the HID protocol in Windows 10, to provide a vendor-specific top-level collection for performing device firmware and vendor configuration updates.

The vendor-specific firmware update collection could provide an output report for transferring the firmware payload from the host to the device. This is highly advantageous, as it allows for firmware updates to be performed without requiring a driver on the host.

It is mandatory for the HID I^2C descriptor member, *wVersionID*, to be incremented after a firmware upgrade.

Integrated Windows pen devices should be able to recover from a firmware update that failed due to power loss (or some other error), by removing, and then reapplying its power supply. It is highly recommended that basic functionality be available even after a failed firmware update.

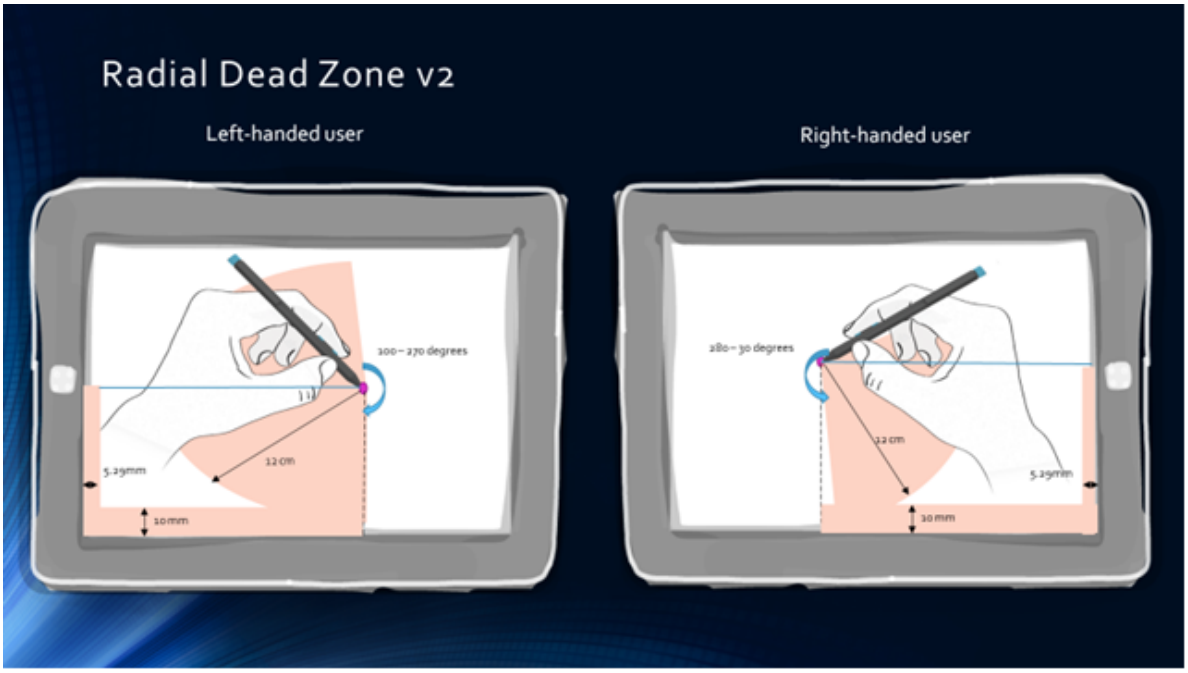
Palm Rejection

Prior to Windows 10, upon receiving input from a pen device that was either in-range or in contact of the screen, the operating system would suppress all touch input (whether in inking or erase mode).

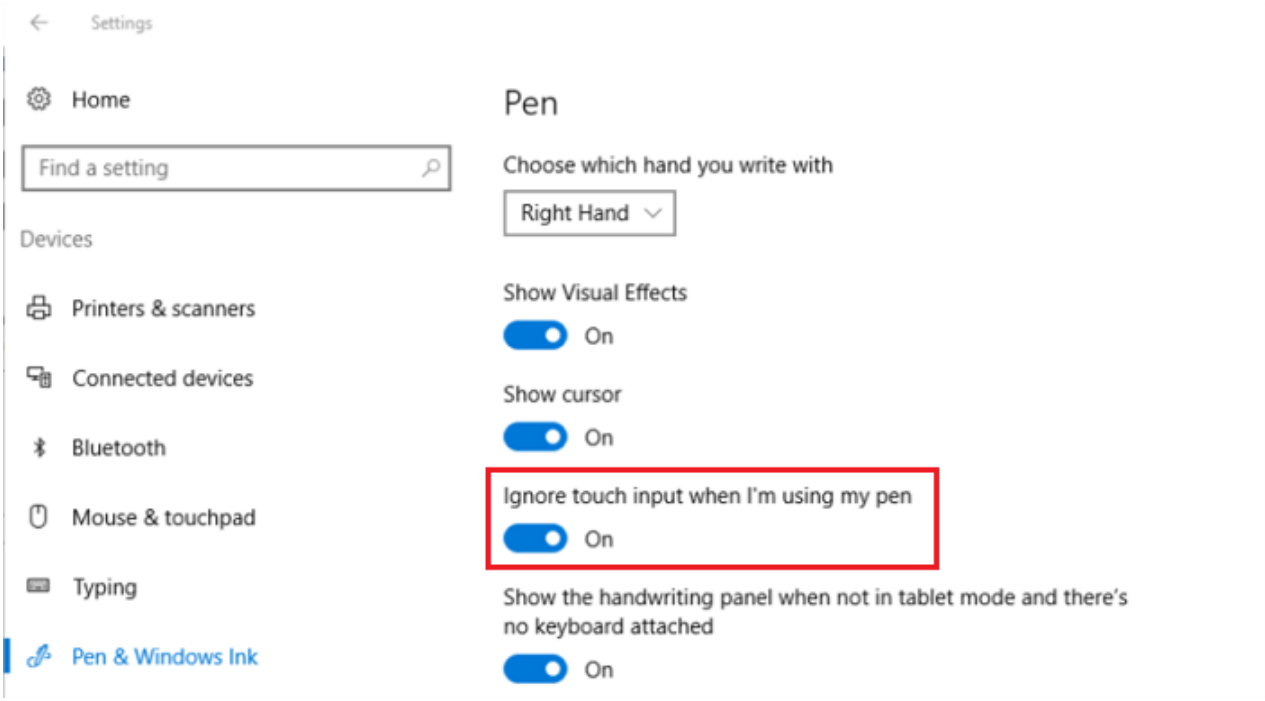
In Windows 10, version 1607, the dead-zone is more compact to aid with simultaneous pen and touch interaction. The modified dead-zone captures almost 95% of inadvertent touches while still providing great end-user experience.

For left-handed users, the dead-zone ranges from 100 - 270 degrees and spans up to 12 cm radially.

For right-handed users, the dead-zone ranges from 280 - 30 degrees and spans up to 12 cm radially.



The operating system accepts touch interactions either from the left or right of the pen tip, depending on the user's setting for handedness. For right handed users, touch interactions are accepted from the left of the pen tip (as shown in the preceding diagram) and vice-versa for left handed users. By default, the pen is configured for right-handed users but can easily be changed by opening **Settings** clicking **Devices**, and then opening the **Pen** dialog window, shown below. However, some users have a preference for the palm rejection behavior found in Windows 8.1. To support this preference, Windows 10 also provides a setting to ignore all touch input when a pen comes in range.



> **Note**
>
> Although legacy devices (specifically ones that suppress touch input when pen input is being delivered) will function in Windows 10, these devices will not deliver the functionality described above. Additionally, these devices will not deliver any future functionality that leverages simultaneous pen and touch reporting.

Starting in Windows 10, this setting can be hidden on systems that do not support simultaneous pen and touch reporting by using the following registry key. Setting the key to 0 will display the settings to the user, while setting it to 1 will hide them.

Registry key	Type	Units
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Pen\HideSPTSettings	DWORD	Boolean (1 or 0)

Touch Interactions with Pen In-Range

As mentioned earlier, Windows 10 allows the user to perform touch interactions while the pen is in-range, on hardware that supports simultaneous touch/pen reporting. But once the pen

transitions from being in-range to being in contact with the screen, all touch interactions are suppressed. This is the same behavior as in previous versions of Windows.

Delayed Zooming on Inking Surfaces

The built-in palm rejection feature in Windows 10, is most effective when the pen is detected in-range of the digitizer, prior to the palm arriving on the digitizer surface. However, depending on the speed at which the palm lands, and the limitation of the pen digitizer's detection threshold, it is possible that the palm will arrive before the pen is detected.

One of the most disruptive results of a palm being reported to the operating system as multiple touch contacts, prior to the pen location being reported is that, an inking surface may change zoom levels inadvertently. Windows 10 provides protection against this behavior by default, by applying a 250ms delay to all pinch/zoom interactions on inking surfaces, to allow additional time for pen detection. It should be noted that this delay will only be present for systems with an integrated pen.

This global delay value (default of 250ms) is configurable by the OEM, via the Windows registry. For pen digitizers that have a reduced detection threshold, where there is strong likelihood of a palm arriving before the pen is detected, it may be advisable to increase the delay value (up to a maximum of 500ms).

For pen digitizers with an improved detection threshold and/or have a touch solution with integrated palm rejection, it may be advisable to reduce this value (down to a minimum of 0ms).

Here's the registry key that can be used to set the pinch/zoom delay value.

Registry key	Type	Units
HKLM\SOFTWARE\Microsoft\Palm\DelayManipulationDuration	DWORD	Milliseconds (ms)

Note that this registry setting and the related delay zoom mechanism are only effective on Windows 10 systems which expose an integrated pen digitizer.

Sample Report Descriptors (pen-sample-report-descriptors)

Article • 12/05/2022

This topic presents a sample report descriptor and some sample descriptor excerpts for a Windows pen device.

> **Note**
>
> Although vendor specific usages may be used at any level of the report, they must be used in a way that could be implemented equally across all reporting methods, namely ,serial, parallel, and hybrid. Not adhering to this guidance may not result in an HLK failure but could result in serious compatibility issues.

Sample Report Descriptor

The following is a report descriptor for an integrated Windows Pen top-level collection.

```
// Integrated Windows Pen TLC
    0x05, 0x0d,                         // USAGE_PAGE (Digitizers)
    0x09, 0x02,                         // USAGE (Pen)
    0xa1, 0x01,                         // COLLECTION (Application)
    0x85, REPORTID_PEN,                 //   REPORT_ID (Pen)
    0x09, 0x20,                         //   USAGE (Stylus)
    0xa1, 0x00,                         //   COLLECTION (Physical)
    0x09, 0x42,                         //     USAGE (Tip Switch)
    0x09, 0x44,                         //     USAGE (Barrel Switch)
    0x09, 0x3c,                         //     USAGE (Invert)
    0x09, 0x45,                         //     USAGE (Eraser Switch)
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //     LOGICAL_MAXIMUM (1)
    0x75, 0x01,                         //     REPORT_SIZE (1)
    0x95, 0x04,                         //     REPORT_COUNT (4)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x95, 0x01,                         //     REPORT_COUNT (1)
    0x81, 0x03,                         //     INPUT (Cnst,Var,Abs)
    0x09, 0x32,                         //     USAGE (In Range)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x95, 0x02,                         //     REPORT_COUNT (2)
    0x81, 0x03,                         //     INPUT (Cnst,Var,Abs)
    0x05, 0x01,                         //     USAGE_PAGE (Generic Desktop)
    0x09, 0x30,                         //     USAGE (X)
    0x75, 0x10,                         //     REPORT_SIZE (16)
```

```
    0x95, 0x01,                    //     REPORT_COUNT (1)
    0xa4,                          //     PUSH
    0x55, 0x0d,                    //     UNIT_EXPONENT (-3)
    0x65, 0x13,                    //     UNIT (Inch,EngLinear)
    0x35, 0x00,                    //     PHYSICAL_MINIMUM (0)
    0x46, 0x3a, 0x20,              //     PHYSICAL_MAXIMUM (8250)
    0x26, 0xf8, 0x52,              //     LOGICAL_MAXIMUM (21240)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x09, 0x31,                    //     USAGE (Y)
    0x46, 0x2c, 0x18,              //     PHYSICAL_MAXIMUM (6188)
    0x26, 0x6c, 0x3e,              //     LOGICAL_MAXIMUM (15980)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0xb4,                          //     POP
    0x05, 0x0d,                    //     USAGE_PAGE (Digitizers)
    0x09, 0x30,                    //     USAGE (Tip Pressure)
    0x26, 0xff, 0x00,              //     LOGICAL_MAXIMUM (255)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x75, 0x08,                    //     REPORT_SIZE (8)
    0x09, 0x3d,                    //     USAGE (X Tilt)
    0x15, 0x81,                    //     LOGICAL_MINIMUM (-127)
    0x25, 0x7f,                    //     LOGICAL_MAXIMUM (127)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x09, 0x3e,                    //     USAGE (Y Tilt)
    0x15, 0x81,                    //     LOGICAL_MINIMUM (-127)
    0x25, 0x7f,                    //     LOGICAL_MAXIMUM (127)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0xc0,                          //   END_COLLECTION
    0xc0                           // END_COLLECTION
```

Sample Report Descriptor Excerpt - (Latency Mode Feature Report)

The following is a report descriptor excerpt from the integrated Windows Pen top-level collection, for the optional (but highly recommended) support of the latency mode feature report.

```
    0x05, 0x0d,                    //    USAGE_PAGE (Digitizer)
    0x85, REPORTID_LATENCY,        //    REPORT_ID (Latency)
    0x09, 0x60,                    //    USAGE(Latency Mode)
    0x75, 0x01,                    //    REPORT_SIZE (1)
    0x95, 0x01,                    //    REPORT_COUNT (1)
    0x15, 0x00,                    //    LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //    LOGICAL_MAXIMUM (1)
    0xb1, 0x02,                    //    FEATURE (Data,Var,Abs)
    0x95, 0x07,                    //    REPORT_COUNT (7)
    0xb1, 0x03,                    //    FEATURE (Cnst,Var,Abs)
```

Sample Report Descriptor Excerpt - (Device Certification Status Feature Report)

The following is a report descriptor excerpt from the integrated Windows Pen top-level collection, for the optional (but highly recommended) support of the device certification status feature report.

```
    0x06, 0x00, 0xff,                   //    USAGE_PAGE (Vendor Defined)
    0x85, REPORTID_PENHQA,              //    REPORT_ID (PTPHQA)
    0x09, 0xC5,                         //    USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //    LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //    LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //    REPORT_SIZE (8)
    0x96, 0x00, 0x01,                   //    REPORT_COUNT (0x100 (256))
    0xb1, 0x02,                         //    FEATURE (Data,Var,Abs)
```

Haptic Pen Implementation Guide

Article • 08/01/2022

This document details the protocol implementation for a Haptic Pen device connecting to a compatible Windows 11 host. This does not include guidance on mechanical constraints, electrical constraints or component selection for generating the haptic response within the pen transducer. This implementation guidance is agnostic of the pen protocol used between the pen transducer and pen digitizer, however an implementation may elect to use a pen protocol with an uplink capability which enables the pen digitizer to provide additional parameters to the pen transducer for the purposes of modulating haptic response.

Device Class

Haptic Pen is an extension of the Pen Device class on Windows. This implementation guide adds to the Pen Implementation Guide and focuses on implementing haptics within the pen transducer, so haptic pens must meet the requirements in Pen Implementation Guide in addition to those contained here.

Device Bus Connectivity

The Haptic Pen shall use the Microsoft provided inbox drivers to connect to a Windows host using HID over Bluetooth.

Haptic Pen Protocol Implementation

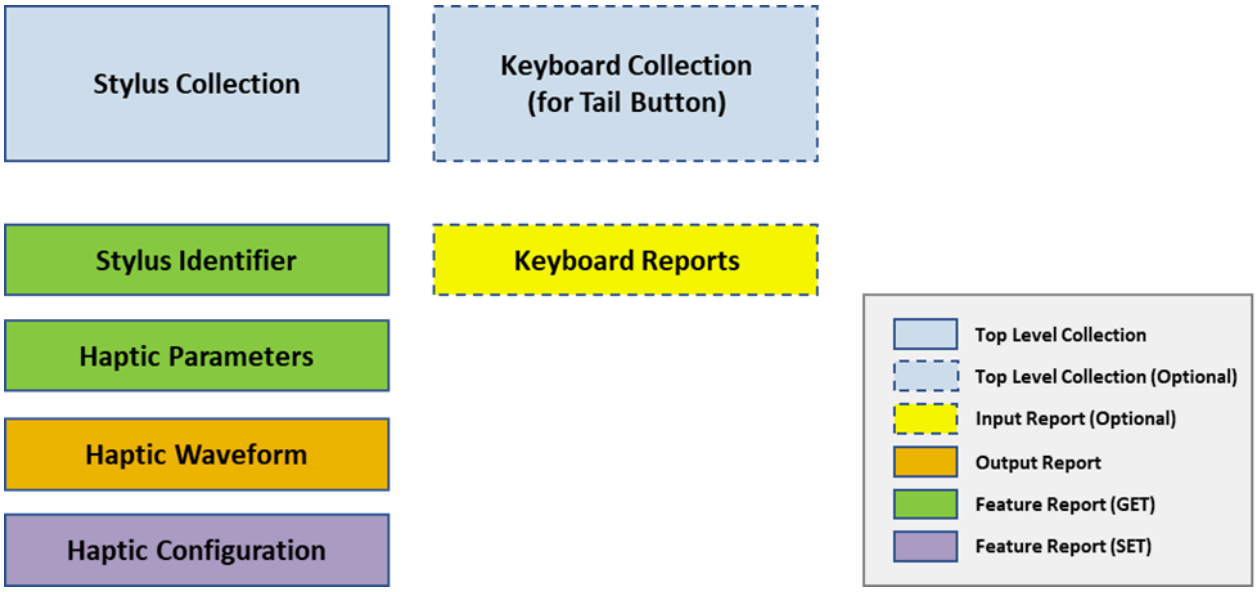
A good understanding of the HID protocol is needed to be able to understand the information presented here. See the following resources for information about the HID protocol:

- Device Class Definition for Human Interface Devices (HID)
- HID Usage Tables

Windows includes a HID class driver and corresponding HID Bluetooth-enabled miniport driver, therefore there is no need for any third-party mini-port drivers. Haptic pen device firmware only needs to report the usages described in this topic. Windows will use the firmware and its own HID drivers to enable the device and give Windows applications access to the device.

A sample descriptor is provided in the Sample Report Descriptors section below.

Required Top Level HID Collection



A Haptic Pen device should use the HID protocol on a Windows 10 system in such a way that the device provides a top-level collection that appears as a digitizer/stylus (Page **0x0D**, Usage **0x20**).

Pen Digitizer Input Report

The pen digitizer collection must report the stylus identifier consisting of the transducer serial number and transducer Vendor ID in the input reports reported to the OS. The same stylus identifier must be reported through the stylus collection. This enables the OS to correlate the pen input generated by the digitizer to the stylus. Details on the pen implementation guide can be found here: Pen Protocol Implementation.

Transducer Serial Number

The Transducer Serial Number is a unique persistent identifier for the transducer used in the pen accessory communicating with the pen digitizer. This is required to be 32-bits and is defined by the vendor or entity identified by the transducer vendor ID. In circumstances where the Transducer Serial Number is unknown to the digitizer either because the pen accessory does not support transmitting this value or that the transmission has not been received in full, the digitizer shall report 0 to the host. Null position is not supported by the host.

Transducer Serial Number – Part 2

Transducer Serial Number – Part 2 allows for an additional 32-bits to be specified as part of the unique persistent identifier for the transducer used in a pen accessory. In

circumstances where the Transducer Serial Number – Part 2 is unknown to the digitizer either because the pen accessory does not support transmitting this value or that the transmission has not been received in full, the digitizer shall report 0 to the host. Null position is not supported by the host.

Expand table

Page	ID	Notes
0xD	0x5B	Mandatory for features dependent on unique pen identification (see below)
0xD	0x6E	Optional extension of the serial number by an additional 32 bits

Transducer Vendor ID

The Transducer Vendor ID is a field for communicating the manufacturer of the transducer used in the pen accessory communicating with the pen digitizer. This is required to be a 2-byte USB-IF assigned vendor ID either of the manufacturer or that of the IHV/OEM authorizing the use of their USB-IF vendor ID for this purpose.

Expand table

Page	ID	Notes
0xD	0x91	Mandatory for features dependent on unique pen identification (see below)

Features Dependent On Unique Pen Identification

Reporting PenID is mandatory to enable scenarios such as pen haptics (required for haptic features in this guide).

It is also mandatory for scenarios which use multiple pens for inking. For example:

- Whiteboard app on windows supports multiple pen usage where each pen can be mapped to a certain inking tool
- In general, apps that want to assign attributes or behaviors to different physical pens, even though the digitizer may only support a single pen on screen at the time
- Apps which want to track multiple simultaneous pens on supported digitizers

Haptic Output Report

If a pen device supports haptic feedback, it can allow the system and applications to take advantage of it by including a haptic feedback collection (Page 0x0E, Usage 0x01) within

the stylus TLC. For more information on how the HID specification supports haptic feedback, please consult the Haptics page ratification to the HID specification.

The host uses the following usages in an output report (through the haptic feedback collection) to allow the host to issue haptic feedback events to the haptic pen device. If a device chooses to expose a haptic feedback collection, some usages are mandatory to allow host-initiated haptic feedback to be supported.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Waveform List	Ordered list of haptic waveforms supported by the device	0x0E	0x10	Mandatory
Duration List	Ordered list of durations for waveforms in the Waveform List	0x0E	0x11	Mandatory
Auto Trigger	Waveform to fire automatically at device's discretion	0x0E	0x20	Mandatory
Auto Trigger Associated Control	HID usage of the control associated with haptic feedback	0x0E	0x22	Mandatory
Intensity	Output - Intensity of Manual Trigger waveform as a percentage	0x0E	0x23	Optional
Repeat Count	Output - Number of times to play Manual Trigger waveform after initial play	0x0E	0x24	Optional
Retrigger Period	Output - Duration of time to wait before re-triggering Manual Trigger when repeating	0x0E	0x25	Optional
Waveform Cutoff Time	Max time a Manual Trigger waveform can play before being cut off	0x0E	0x28	Optional

Waveform List

Waveform List usage represents a collection of the HID usages of supported waveforms, ordered using ordinals. The pre-defined haptics waveforms are defined in the HID specification. For pen haptic devices, these waveforms can be classified as two segments that correspond to different scenarios:

- WAVEFORM_*CONTINOUS - Ink based feedback for simulating various textures while the user is actively inking with different tools such as pen, pencil etc.

- WAVEFORM_* - Discrete, non-continuous interaction-based feedback for when a user performs some input driven task such as hovering over a button, clicking a disabled button, and successful ink shape recognition.

The full list of waveforms supported for pen haptic devices is below:

Expand table

Waveform	Description	Page	ID	Mandatory/Optional
WAVEFORM_NONE	No-op. Should not impact the play state of ongoing waveforms	0x0E	0x1001	Mandatory
WAVEFORM_STOP	Stops the playback of ongoing waveforms	0x0E	0x1002	Mandatory
WAVEFORM_CLICK	Creates a short “click” feedback. Default fallback when interaction feedback waveform selected by the app is not supported by haptic pen	0x0E	0x1003	Mandatory
WAVEFORM_INKCONTINUOUS	Simulates the feel of inking with a physical ball point pen. Default fall back when an inking waveform is not supported by haptic pen	0x0E	0x100B	Mandatory
WAVEFORM_SUCCESS	Strong haptic signal to alert user an action has succeeded	0x0E	0x1009	Optional
WAVEFORM_ERROR	Strong haptic signal to alert user an action has failed, or an error has occurred	0x0E	0x100A	Optional

Waveform	Description	Page	ID	Mandatory/Optional
WAVEFORM_HOVER	Haptic signal when user hovers over an interactive UI element with a haptic pen	0x0E	0x1008	Optional
WAVEFORM_PRESS	Haptic signal for when a user presses an interactive UI element in an incremental action (see Release)	0x0E	0x1006	Optional
WAVEFORM_RELEASE	Haptic signal for when a user releases an interactive UI element in an incremental action (see Press)	0x0E	0x1007	Optional
WAVEFORM_PENCILCONTINUOUS	Continuous haptic signal when user selects pencil as inking tool	0x0E	0x100C	Optional
WAVEFORM_MARKERCONTINUOUS	Continuous haptic signal when user selects marker as inking tool	0x0E	0x100D	Optional
WAVEFORM_CHISELMARKERCONTINUOUS	Continuous haptic signal when user selects chisel marker/highlighter as inking tool	0x0E	0x100E	Optional
WAVEFORM_BRUSHCONTINUOUS	Continuous haptic signal when user selects brush as inking tool	0x0E	0x100F	Optional
WAVEFORM_ERASERCONTINUOUS	Continuous haptic signal when user selects eraser as inking tool	0x0E	0x1010	Optional

Waveform	Description	Page	ID	Mandatory/Optional
WAVEFORM_SPARKLECONTINUOUS	Continuous haptic signal for special ink tools, such as a multi-colored brush	0x0E	0x1011	Optional

> **Note**
>
> While not required, it is encouraged to also implement the other enumerated waveforms in order to provide a more complete user experience. In particular, WAVEFORM_PRESS and WAVEFORM_RELEASE are highly recommended as they provide valuable interaction feedback.

Both WAVEFORM_NONE and WAVEFORM_STOP are required for all HID-compliant haptics devices. Ordinals 1 and 2 are implicitly set to WAVEFORM_NONE and WAVEFORM_STOP. They do not need to be declared in the Waveform List or Duration List. The Waveform List declares supported waveforms by the physical minimum and maximum of each ordinal in the list.

Duration List

Duration List usage represents a collection of the durations for the supported waveforms in the Waveform list, ordered using ordinals. The unit for waveform duration is milliseconds, and duration must be a positive non-zero value for any non-continuous waveform. If a waveform is continuous (will play until stopped by the host or the Waveform Cutoff Time is exceeded), then its duration is defined as zero.

WAVEFORM_NONE and WAVEFORM_STOP are assumed to have a duration of zero. They do not need to be declared in the Duration List.

Intensity

The Intensity usage represents the percentage of maximum intensity to apply to a waveform. This value should vary between 0 and 100 percent. 100 percent indicates waveforms will be triggered by the device at their maximum strength, and 0 percent indicates the haptic transducer is not enabled.

Repeat Count

The Repeat Count usage represents the number of times to repeat a waveform. A Repeat Count of zero indicates the Manual Trigger waveform should only be played a single time (no repeat). If the Waveform Cutoff Time has been exceeded, it is expected that any incomplete repeats will be ignored.

Retrigger Period

The Retrigger Period usage represents the amount of time for the device to wait before repeating a Manual Trigger waveform in an output report, per the value specified by Repeat Count. The units for this value are milliseconds. If the Retrigger Period is smaller than the duration of the waveform being played, the waveform should be stopped and restarted at the time period indicated by the retrigger period.

Waveform Cutoff Time

The Waveform Cutoff Time usage represents the maximum amount of time the device will allow a Manual Trigger waveform to be repeated before ending playback. This is a constant value for the device and includes both continuous waveforms without set durations, and waveforms with discrete durations which are set to repeat many times. The units for this value are milliseconds.

Haptic Output Report

The host uses the following usages in an output report to issue haptic feedback events to the Haptic Pen device. Some usages are Mandatory for compatibility with the Windows host implementation.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Manual Trigger	Waveform to fire as explicit command from the host	0x0E	0x21	Mandatory
Intensity	Intensity of Manual Trigger waveform as a percentage	0x0E	0x23	Mandatory
Repeat Count	Number of times to play Manual Trigger waveform after initial play	0x0E	0x24	Optional
Retrigger Period	Duration of time to wait before retriggering Manual Trigger when repeating	0x0E	0x25	Optional

Manual Trigger

The Manual Trigger usage represents a supported waveform usage from the waveform list which has been requested to be played by the host. When an output report containing a Manual Trigger other than WAVEFORM_NONE is sent to the device, it should immediately begin playing the specified waveform with the additional properties included in the output report (Intensity, Repeat Count, Retrigger Period). When an output report contains a Manual Trigger of WAVEFORM_STOP, any ongoing waveform playback should be stopped.

For **Intensity**, **Repeat Count**, and **Retrigger Period** usages, please see the previous section with respect to the output feature report.

Starting and Stopping Haptics

The flowchart below describes when the pen's haptic signals should be configured, cleared, started, and stopped.

The various haptics states described below are:

- **Playing**: The pen is actively playing the haptic waveform
- **Paused**: The pen is configured with a waveform, but is not actively playing it
- **Stopped**: The pen is not configured with a waveform and is not actively playing anything

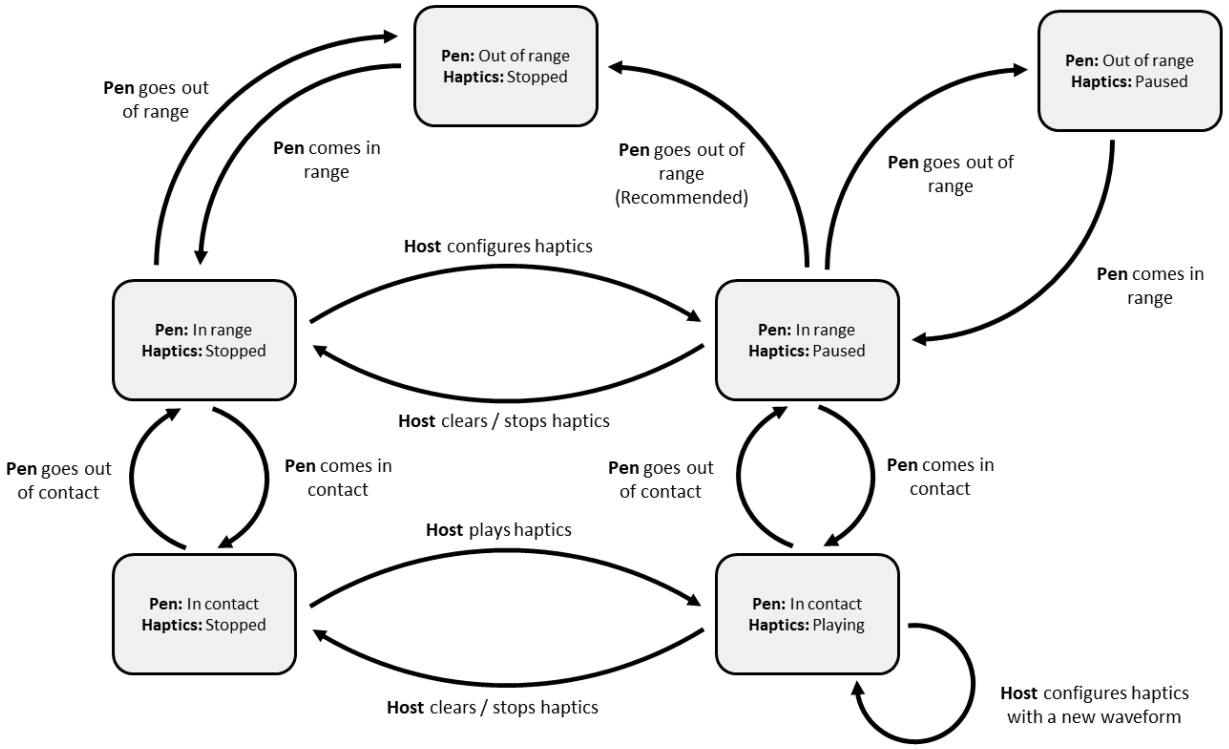
For the pen state with respect to the digitzer, please refer to Windows Pen States.

> ⓘ **Note**
>
> When the pen goes out of range it's recommended, but not required, to clear the haptic configuration. This is conveyed in the diagram below by the two alternate paths exiting from the "Pen: In range; Haptics: Paused" state when the pen goes out of range.

> ⓘ **Note**
>
> The host at any time may request for a non-continuous waveform to be played. In that event, the pen should play it and then return to the state it was previously in.

> ⓘ **Note**

The host shall only configure continuous waveforms. Discrete/non-continuous waveforms shall only be manually triggered.



Keyboard Collection (Optional)

An optional feature to enable reporting of tail-end button clicks to the host via HID keyboard reports.

A compatible device shall report 3 distinct keyboard combinations corresponding to 3 distinct button actions via a HID Bluetooth keyboard device exposed to the host. The actions and corresponding keyboard combinations are outlined below:

Expand table

Button Action	Key Combination
Single Click	WIN+F20
Double Click	WIN+F19
Press and Hold	Win+F18

Bluetooth Button Implementation

To implement a tail-end Bluetooth button, the device will report 3 distinct keyboard combinations corresponding to 3 distinct button actions via a HID Bluetooth LE keyboard

device exposed to the host. The actions and corresponding keyboard combinations are outlined below:

Expand table

Bluetooth Button Action	Key Combination to Report
Single-click	WIN+F20
Double-click	WIN+F19
Press and hold	WIN+F18

Pen Stowage

Starting with Windows 10, version 1903, Windows supports notifications for devices incorporating compatible pen stowage. The mechanism relies on the hardware detecting the pen being removed or replaced and generating a corresponding HID keyboard report for a pair of shortcut combinations. To signal a dock (pen replaced in stowage), report **WIN+CTRL+F20** and to signal an undock (pen removed from stowage), report **WIN+CTRL+F19**. This can be implemented with firmware or a driver.

These undock/dock events bring-up/dismiss the Shell ink workspace menu. Starting with Windows 10, version 2004 Office also reacts to these events by using a platform API that enables any developer to extend their application for awareness of stowage events. There is no support to query whether the pen is present in the dock, apps are only notified of removal and return events if they are in the foreground.

Sample HID Report Descriptor

```
05,0D,                    // Usage Page (Digitizers)
09,20,                    // Usage (Stylus)
A1,01,                    // Collection (Application)
85,40,                    //   Report ID (64)
95,01,                    //   Report Count (1)
75,20,                    //   Report Size (32)
17,00,00,00,80,           //   Logical Minimum (-2147483648)
27,FF,FF,FF,7F,           //   Logical Maximum (2147483647)
09,5B,                    //   Transducer Serial Number
81,02,                    //   Input (Data,Var,Abs)
75,10,                    //   Report Size (16)
15,01,                    //   Logical Minimum (1)
27,FF,FF,00,00,           //   Logical Maximum (65535)
09,91,                    //   Transducer Vendor ID
81,02,                    //   Input (Data,Var,Abs)
```

```
05,0E,                    //   Usage Page (Haptics)
09,01,                    //   Usage (0x01)
A1,02,                    //   Collection (Logical)
85,41,                    //     Report ID (65)
95,01,                    //     Report Count (1)
75,08,                    //     Report Size (8)
15,01,                    //     Logical Minimum (1)
26,FF,00,                 //     Logical Maximum (255)
09,24,                    //     Usage (0x24)
B1,02,                    //     Feature (Data,Var,Abs)
09,24,                    //     Usage (0x24)
91,02,                    //     Output (Data,Var,Abs)
09,23,                    //     Usage (0x23)
B1,02,                    //     Feature (Data,Var,Abs)
09,23,                    //     Usage (0x23)
91,02,                    //     Output (Data,Var,Abs)
15,01,                    //     Logical Minimum (1)
25,12,                    //     Logical Maximum (18)
09,20,                    //     Usage (0x20)
B1,02,                    //     Feature (Data,Var,Abs)
09,21,                    //     Usage (0x21)
91,02,                    //     Output (Data,Var,Abs)
15,00,                    //     Logical Minimum (0)
26,FE,00,                 //     Logical Maximum (254)
66,01,10,                 //     Unit (SI Linear)
55,FD,                    //     Unit Exponent (253)
35,00,                    //     Physical Minimum (0)
46,EC,09,                 //     Physical Maximum (2540)
09,28,                    //     Usage (0x28)
91,02,                    //     Output (Data,Var,Abs)
75,10,                    //     Report Size (16)
26,D0,07,                 //     Logical Maximum (2000)
46,D0,07,                 //     Physical Maximum (2000)
09,25,                    //     Usage (0x25)
91,02,                    //     Output (Data,Var,Abs)
09,25,                    //     Usage (0x25)
B1,02,                    //     Feature (Data,Var,Abs)
45,00,                    //     Physical Maximum (0)
85,42,                    //     Report ID (66)
75,20,                    //     Report Size (32)
17,42,00,0D,00,           //     Logical Minimum (852034)
27,42,00,0D,00,           //     Logical Maximum (852034)
09,22,                    //     Usage (0x22)
B1,02,                    //     Feature (Data,Var,Abs)
09,11,                    //     Usage (0x11)
A1,02,                    //     Collection (Logical)
05,0A,                    //       Usage Page (Ordinal)
75,10,                    //       Report Size (16)
95,10,                    //       Report Count (16)
15,01,                    //       Logical Minimum (1)
27,FF,FF,00,00,           //       Logical Maximum (65535)
19,03,                    //       Usage Minimum (0x03)
29,12,                    //       Usage Maximum (0x12)
B1,02,                    //       Feature (Data,Var,Abs)
C0,                       //     End Collection ()
```

```
05,0E,                          //     Usage Page (Haptics)
09,10,                          //     Usage (0x10)
A1,02,                          //     Collection (Logical)
05,0A,                          //       Usage Page (Ordinal)
16,03,10,                       //       Logical Minimum (4099)
26,FF,2F,                       //       Logical Maximum (12287)
19,03,                          //       Usage Minimum (0x03)
29,12,                          //       Usage Maximum (0x12)
B1,02,                          //       Feature (Data,Var,Abs)
C0,                             //     End Collection ()
C0,                             //   End Collection ()
C0                              // End Collection ()
```

Seamless Pairing for Windows Pens Implementation Guide

Article • 07/13/2023

> **Note**
>
> This document details the protocol implementation to enable seamless Bluetooth pairing for compatible transducers via a Pen digitizer on Windows 11. This implementation guidance is agnostic of the pen protocol used between the pen transducer and pen digitizer however an implementation may elect to use any pen protocol with an uplink capability to facilitate the exchange of the relevant parameters herein described.

Device Class

Seamless Bluetooth pairing support is an extension of the Pen Device class on Windows. This implementation guide adds to the existing Pen Devices implementation guide and focuses on implementing necessary parameters exchange between the host and pen transducer.

Device Bus Connectivity

Please consult the prerequisite guidance here: Protocol Implementation (pen-protocol-implementation) | Microsoft Docs

A good understanding of the HID protocol is needed to be able to understand the information presented here. See the following resources for information about the HID protocol:

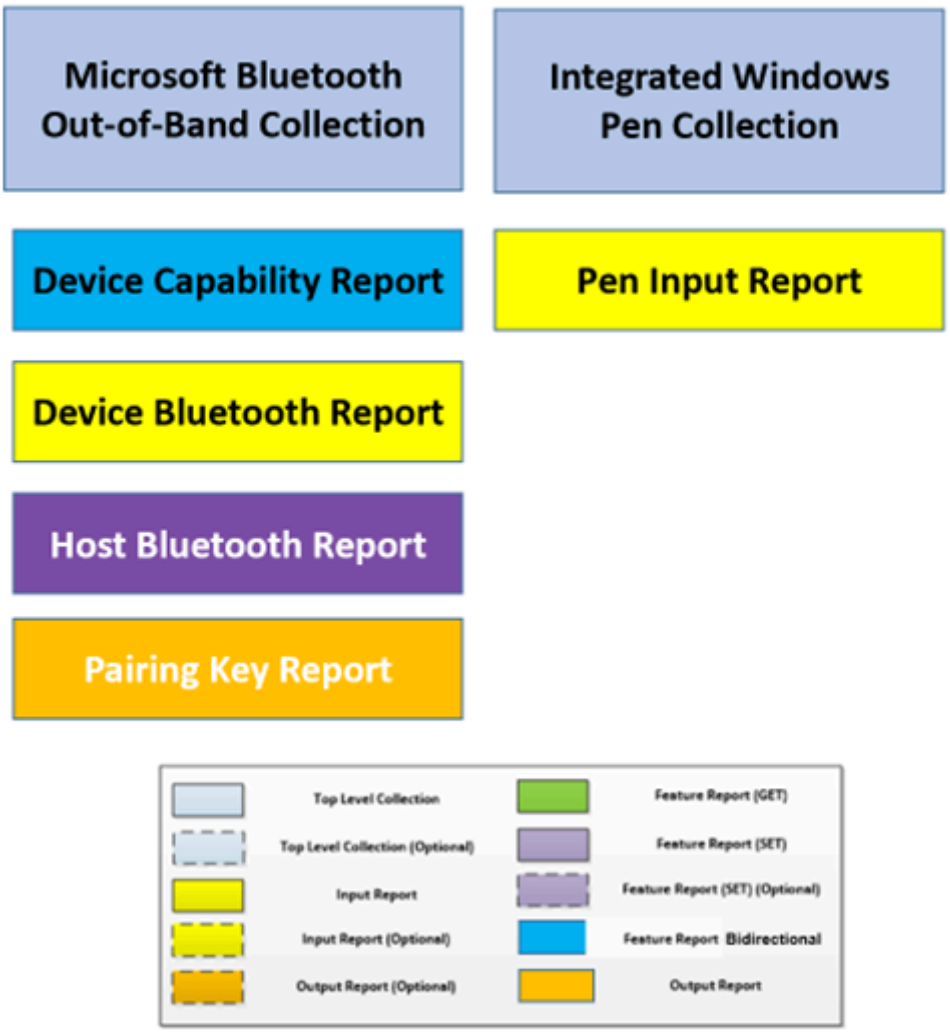
- Device Class Definition for Human Interface Devices (HID)
- HID Usage Tables

Windows includes a HID class driver and corresponding miniport drivers HID I^2C, HID SPI, HID USB and HID BTH miniport drivers, therefore there is no need for any third-party mini-port drivers.

The pen digitizer firmware only needs to report the usages described in this topic. Windows will leverage compatible firmware and its own inbox drivers to support the seamless pairing experience.

A sample descriptor is provided in the Sample Report Descriptors section below.

Required Top Level HID Collections



A pen digitizer capable of supporting seamless pairing shall provide a top-level collection (TLC) that appears as vendor specific (Page **0xFFF4**, Usage **0x01**). This shall be a sibling TLC to the Integrated Windows Pen Collection.

INF Decorations for Digitizer Opt-in to Seamless Pairing

Given the use of Microsoft vendor specific HID usages in this implementation guide, it is a prerequisite for a pen digitizer to explicitly opt-in to allowing the OS to communicate with the corresponding TLC exposing them. The below should be furnished in an equivalent extension INF corresponding to the pen digitizer device. The presence of the device property below will signal that opt-in.

[Digitizer_Device.NT.Interfaces] AddInterface = {4D1E55B2-F16F-11CF-88CB-001111000030},,Digitizer_Device.AddInterface

[Digitizer_Device.AddInterface] AddProperty = Digitizer_Device.AddProperty

[Digitizer_Device.AddProperty] ; DEVPKEY_BluetoothOobCouplingChannel_OptInForUse {FE683ED0-DE80-4766-B422-4195993F4330}, 2, 0x11, , 1

Device Capability Report (Usage 0x16)

This is a bidirectional feature report meaning the host shall convey its capability to the digitizer on enumeration and subsequently query the digitizer's capability. The following usage shall be present in the Device Capability Report to convey that the digitizer or host is able to support seamless pair capable pen transducers.

This feature report will be sent and subsequently retrieved upon enumeration.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Seamless Pair Capable	Indicates whether the digitizer is able to support seamless pair capable pen transducers (bit flag where 1 indicates capability)	0xFFF4	0x10	Mandatory

Device Bluetooth Report (Usage 0x13)

The following usages shall be present in the Device Bluetooth Report which conveys relevant parameters about the Bluetooth radio integrated with the pen transducer.

This input report is only sent one time after a seamless pair capable transducer is in-range and verification of whether an existing pairing relationship with the host has been completed.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Device Bluetooth Address	The 48-bit Bluetooth MAC address associated with the radio in the transducer	0xFFF4	0x02	Mandatory
Device Bluetooth Address Type	Indicates whether the Device Bluetooth Address is not public	0xFFF4	0x08	Mandatory
Host Pairing Exists	Indicates whether the transducer already has an existing pairing with	0xFFF4	0x11	Mandatory

Member	Description	Page	ID	Mandatory/Optional
	this host			

Mandatory Usages

Device Bluetooth Address

This usage is expected to be 6 bytes containing the 48-bit MAC address associated with the Bluetooth radio integrated into the pen transducer. i.e. 11:22:33:44:55:66

Device Bluetooth Address Type

This usage is expected to be a flag indicating whether the Device Bluetooth Address usage should be interpreted by the host as public indicated by the flag clear = 0 or random indicated by the flag set = 1.

Seamless Pair Capable Transducer In-Range

This usage is expected to be a flag indicating whether a transducer capable of seamless pairing is now in range of the pen digitizer indicated TRUE by the flag set = 1. If the transducer is out-of-range or the transducer that is in-range isn't capable of seamless pairing the entire input report should not be generated.

Host Pairing Exists

This usage is expected to be a flag indicating whether the seamless capable transducer which is in-range already has a paring relationship with this host. This is indicated to be TRUE by the flag set = 1. If there is no pairing relationship with this host this flag should be clear = 0.

Host Bluetooth Report (Usage 0x06)

The following usages shall be present in the Host Bluetooth Report which conveys relevant parameters about the hosts Bluetooth radio.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Host Bluetooth Address	The 48-bit Bluetooth MAC address associated with the radio of the host	0xFFF4	0x03	Mandatory
Host Bluetooth Address Type	Indicates whether the Host Bluetooth Address is public	0xFFF4	0x09	Mandatory

Mandatory Usages

Host Bluetooth Address

This usage is expected to be 6 bytes containing the 48-bit MAC address associated with the Bluetooth radio of the host. i.e. 11:22:33:44:55:66

Host Bluetooth Address Type

This usage is expected to be a flag indicating whether the Host Bluetooth Address usage should be interpreted by the digitizer and transducer as public indicated by the flag clear = 0 or random indicated by the flag set = 1. Note: On Windows, this shall always be of type public with the flag clear = 0.

Pairing Key Report (Usage 0x15)

Expand table

Member	Description	Page	ID	Mandatory/Optional
Host Pairing Key	The 128-bit pairing key provided by the host	0xFFF4	0x14	Mandatory
Device Bluetooth Address	The 48-bit Bluetooth MAC address associated with the radio in the transducer	0xFFF4	0x02	Mandatory
Device Bluetooth Address Type	Indicates whether the Device Bluetooth Address is not public	0xFFF4	0x08	Mandatory

Mandatory Usages

Host Pairing Key

This usage is expected to be 16 bytes containing the 128-bit temporary pairing key issued by the host to facilitate the seamless pairing of the previously reported capable transducer. If the host is locked/secured or isn't seamless pairing capable, the pairing key report shall never be issued by the host.

Device Bluetooth Address

This is the same usage per the Device Bluetooth Report above, however this is used by the host to indicate which Bluetooth device the Host Pairing Key in this report is associated with in the scenarios where a digitizer supports multiple Bluetooth capable pens simultaneously.

Device Bluetooth Address Type

This is the same usage per the Device Bluetooth Report above and used in conjunction with Device Bluetooth Address as part of the indexing for the Host Pairing Key.

Expected Sequence

1. Enumeration

On a Windows 10 or Windows 11 OS, the Microsoft [Vendor Specific] Bluetooth Out-of-band Collection will be opened by a built in OS service once enumerated. The host will then issue a SET FEATURE for the Host Bluetooth Report with the Host Bluetooth Address and Host Bluetooth Address Type. The host will then also issue a GET FEATURE for the Device Capability Report. If the digitizer doesn't return this report or returns 0, no further actions related to seamless pairing shall occur.

2. Seamless Capable Pen In-Range

Once a seamless capable pen has come in-range of the digitizer and whether any existing pairing relationship with the Host Bluetooth address has been ascertained then a Device Bluetooth input report shall be issued one time with all mandatory usages.

> ⓘ **Note**
>
> If the pen which has come in-range of the digitizer is not seamless capable, no input report shall be issued. If the seamless pair capable pen goes out-of-range, there is no expectation of an out-of-range indication via this report. The state

machine for a given seamless capable device shall reset upon each receipt of an input report with its Bluetooth address.

The mechanism for how the digitizer firmware obtains the parameters involving the Device Bluetooth Address from the transducer is pen protocol specific and intrinsically defined by the respective specification (i.e. MPP 2.6, AES 2.1)

3. Host Pairing Key Generation

If the Device Bluetooth report indicates that the pen in-range doesn't have an existing pairing relationship with this host then provided that the host is not locked/secure, a temporary pairing key shall be generated per the Device Bluetooth Address. The host shall then issue the Pairing Key output report that contains the Host Pairing Key.

> ⓘ **Note**
>
> On Windows 10, the OS service does not support seamless pairing and therefore will never provide a Host Pairing Key and will therefore never issue this output report.

The mechanism for how the digitizer firmware provides the temporary pairing key to the transducer's Bluetooth radio is pen protocol specific and intrinsically defined by the respective specification (i.e. MPP 2.6, AES 2.1)

4. Pairing

Once the host has issued the output report with the pairing key, it will be waiting up to 30 seconds for advertisement. The pen's Bluetooth connection shall then send a connectable Directed Advertisement to the host Bluetooth radio to form a connection which will then allow the host to initiate the pairing ceremony specifying the OOB (out-of-band) method. At this point the pen's Bluetooth radio will verify the host address was the same it received (via the SET FEATURE report) and then provide the appropriate pairing response including the temporary key. The host Bluetooth radio will verify the temporary key and upon success the pen and host will exchange session encryption keys and complete the pairing ceremony.

If the pairing is successful, the next time the pen comes back into range, the Device Bluetooth input report shall indicate that Host Pairing Exists = 1 (TRUE).

Sample Report Descriptor

```
//Microsoft Bluetooth Out-of-Band TLC
0x06, 0xF4, 0xFF, // Usage Page (Vendor Defined 0xFFF4)
0x09, 0x01, // Usage (0x01)
0xA1, 0x01, // Collection (Application)

0x09, 0x16, // USAGE (Device Capability Report)
0xA1, 0x02, // COLLECTION (Logical)
0x85, 0x70, // REPORT_ID (0x70)
0x09, 0x10, // USAGE (Seamless Pair Capable)
0x75, 0x01, // REPORT_SIZE (1)
0x95, 0x01, // REPORT_COUNT (1)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0x01, // LOGICAL_MAXIMUM (1)
0xB1, 0x02, // FEATURE (Data,Var,Abs)
0x75, 0x07, // REPORT_SIZE (7)
0x95, 0x01, // REPORT_COUNT (1)
0xB1, 0x01, // FEATURE (Const,Var,Abs)
0xC0, // END_COLLECTION

0x09, 0x06, // USAGE (Host Bluetooth Report)
0xA1, 0x02, // COLLECTION (Logical)
0x85, 0x56, // REPORT_ID (0x56)
0x09, 0x03, // USAGE (Host Bluetooth Address)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0xFF, // LOGICAL_MAXIMUM (255)
0x75, 0x08, // REPORT_SIZE (8)
0x95, 0x06, // REPORT_COUNT (6)
0xB2, 0x02, 0x01, // FEATURE (Data,Var,Abs,Buf)
0x09, 0x09, // USAGE (Host Bluetooth Address Type)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0x01, // LOGICAL_MAXIMUM (1)
0x75, 0x01, // REPORT_SIZE (1)
0x95, 0x01, // REPORT_COUNT (1)
0xB1, 0x02, // FEATURE (Data,Var,Abs)
0x75, 0x07, // REPORT_SIZE (7)
0x95, 0x01, // REPORT_COUNT (1)
0xB1, 0x01, // FEATURE (Cnst,Ary,Abs)
0xC0, // END_COLLECTION

0x09, 0x13, // USAGE (Device Bluetooth Report)
0xA1, 0x02, // COLLECTION (Logical)
0x85, 0x6E, // REPORT_ID (0X6E)
0x09, 0x02, // USAGE (Device Bluetooth Address)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0xFF, // LOGICAL_MAXIMUM (255)
0x75, 0x08, // REPORT_SIZE (8)
0x95, 0x06, // REPORT_COUNT (6)
0x82, 0x02, 0x01, // INPUT (Data,Var,Abs,Buf)
0x09, 0x08, // USAGE (Device Bluetooth Address Type)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0x01, // LOGICAL_MAXIMUM (1)
0x75, 0x01, // REPORT_SIZE (1)
```

```
0x95, 0x01, // REPORT_COUNT (1)
0x81, 0x02, // INPUT (Data,Var,Abs)
0x09, 0x11, // USAGE (Host Pairing Exists)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0x01, // LOGICAL_MAXIMUM (1)
0x75, 0x01, // REPORT_SIZE (1)
0x95, 0x01, // REPORT_COUNT (1)
0x81, 0x02, // INPUT (Data,Var,Abs)
0x75, 0x06, // REPORT_SIZE (6) [PADDING]
0x95, 0x01, // REPORT_COUNT (1)
0x81, 0x01 // INPUT (Cnst,Ary,Abs)
0xC0, // END_COLLECTION


0x09, 0x15, // USAGE (Pairing Key Report)
0xA1, 0x02, // COLLECTION (Logical)
0x85, 0x6f, // REPORT_ID (0X6F)
0x09, 0x02, // USAGE (Device Bluetooth Address)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x26, 0xff, 0x00, // LOGICAL_MAXIMUM (255)
0x75, 0x08, // REPORT_SIZE (8)
0x95, 0x06, // REPORT_COUNT (6)
0x92, 0x02, 0x01, // OUTPUT (Data,Var,Abs,Buf)
0x09, 0x08, // USAGE (Device Bluetooth Address Type)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0x01, // LOGICAL_MAXIMUM (1)
0x75, 0x01, // REPORT_SIZE (1)
0x95, 0x01, // REPORT_COUNT (1)
0x91, 0x02, // OUTPUT (Data,Var,Abs)
0x75, 0x07, // REPORT_SIZE (7)
0x95, 0x01, // REPORT_COUNT (1)
0x91, 0x01, // OUTPUT (Cnst,Ary,Abs)
0x09, 0x14, // USAGE (Temporary Key)
0x15, 0x00, // LOGICAL_MINIMUM (0)
0x25, 0xFF, // LOGICAL_MAXIMUM (255)
0x75, 0x08, // REPORT_SIZE (8)
0x95, 0x10, // REPORT_COUNT (16)
0x92, 0x02, 0x01, // OUTPUT (Data,Var,Abs,Buf)
0xC0, // END_COLLECTION
```

Digitizer Display Mapping

Article • 05/20/2024

This section describes how a Windows touch or pen ditizer can be automatically mapped to the appropriate display, for Windows 10 and later operating systems.

The approach for automatically mapping the digitizer and its display differs for internal vs. external displays.

External displays

Windows will match external digitizers to their display using their Container ID configuration. Windows automatically generates a unique Container ID for the display by using the manufacturer ID, product ID, and serial number obtained from the Extended Display Identification Data (EDID). The display may override this Container ID by handling the DXGKDDI_GET_CHILD_CONTAINER_ID callback. The digitizer must then use this same Container ID, typically via its USB ContainerID descriptor.

If there is a single external display, the only requirement is that the digitizer is also marked as external - no other properties need to match. Matching by Container ID is only necessary when there are multiple external displays, even if not all of them are touch/pen-enabled.

> **Note**
>
> If the user has multiple touch/pen-enabled displays of the same make and model connected, it is important that each display has a different Container ID, to ensure that each of the digitizers is properly mapped to the appropriate display. If multiple devices share the same Container ID, Windows will be unable to map them. If the Container ID is derived from the EDID, then the EDID must have a unique serial number for every device.

Internal displays

If there is a single internal display, the only requirement is that the digitizer is also marked as internal - no other properties need to match.

When there are multiple internal displays, Windows will match digitizers to their display using their ACPI Physical Location Description (_PLD) configuration. In order for the

mapping to occur, the digitizer and display must match:

1. Their container IDs (typically this is the "internal" container ID)
2. Their _PLD **CabinetNumber** value (or if only _PLD v1 is supported, their **Lid** and **Dock** values)
3. Their _PLD **Panel** value

User overrides

Windows allows a user to override their digitizer's display mapping by opening Control Panel > Hardware and Sound > Tablet PC Settings, then clicking **Setup...**.

Related topics

Container IDs

Container ID support for displays

Using Microsoft OS Container ID Descriptors

DXGKDDI_GET_CHILD_CONTAINER_ID callback function

ACPI _PLD Specification

Pen Validation Guide

Article • 11/05/2021

This section is a validation guide for an integrated Pen device (a Windows Active Pen), for Windows 10 and later operating systems.

The guidance provided here ensures compliance with Hardware Lab Kit (HLK) testing for a Windows Active Pen.

In this section

Topic	Description
Pen Validation Overview	This topic presents a brief overview of the assumed device test conditions for pen validation, and outlines critical pre-test requirements.
Hardware Requirements and Vendor Information	This topic provides a list of the test equipment that is required for the pen-related tests in the Windows Hardware Lab Kit (HLK), and information about the vendors who stock the listed equipment.
General Testing Guidelines	This topic provides general guidelines for testing a Windows pen device, using the Windows Hardware Lab Kit (HLK) for Windows 10
Pen Testing Interface	This topic presents the user interface for the pen tests in the Windows Hardware Lab Kit (HLK) for Windows 10
Active Pen Tests	This section lists the topics that provide details about the active pen tests in the Windows Hardware Lab Kit (HLK) for Windows 10.
Simultaneous Pen and Touch Validation	This section provides details on validating simultaneous pen and touch reporting for Windows 10
Haptic Pen Validation Guide	This section provides details on validating haptic pen features.
Appendix	The appendix for the Windows Pen Validation Guide, provides information about manually running the Windows Hardware Lab Kit (HLK) tests, some points to note (following our latest content release), and answers to some frequently asked questions.

Pen Validation Overview

Article • 01/18/2019

This topic presents a brief overview of the assumed device test conditions for pen validation, and outlines critical pre-test requirements.

In order to ensure that all Windows Hardware Lab Kit (HLK) tests run successfully, it is important to follow the guidelines in this document as closely as possible.

Device Test conditions

This Pen Validation Guide makes the following assumptions about the testing conditions of the device:

- HLK tests are being run on a device with Windows 10 installed.

  Only the Windows 10 for desktop editions (Home, Pro, Enterprise, and Education) are supported by the HLK tests at this time. See the Appendix: Frequently Asked Questions section for additional information.

- Device under test has a diagonal screen size of 4.5" - 30."

  See the Appendix: Frequently Asked Questions section for additional information.

- Device reports HID usages as described in the Pen Implementation Guide.

- Testing setup has all the required equipment detailed in Hardware Requirements and Vendor Information.

- For each HLK test, the user follows the General Testing Guidelines, as well as the notes and instructions that are specific to the test - as outlined in Active Pen Tests.

If your device or testing setup does not meet the above criteria, you will not be able to run and pass the Windows 10 HLK tests.

Critical Pre-Testing Requirement

Note Failure to follow these steps will cause tests to crash on launch, or fail to receive input.

These steps must be performed on the device under test before running the pen HLK tests.

- Put the device into Developer Mode:

  To put your device into Developer Mode, navigate to the **Settings** > **Update & security** dialog window. Then select **For developers** in the left-hand pane, and click **Developer Mode** in the right-hand window.

- Enable test signing:

  To enable test signing, open an Administrator Command Prompt and Enter the command "**bcdedit /set testsigning on**", then restart the system.

The remaining topics in the Pen Validation Guide, provide more detailed information about the setup and other procedures for performing the pen-related tests in the HLK.

Hardware Requirements and Vendor Information (hardware-requirements-and-vendor-information)

Article • 02/24/2023

This topic provides a list of the test equipment that is required for the pen-related tests in the Windows Hardware Lab Kit (HLK), and information about the vendors who stock the listed equipment.

Here's a list of all the required test equipment for completing the Active Pen Tests in the HLK. The images, instructions and examples in Active Pen Tests are based on the original TRIEX tools, but you should be able to run all the tests successfully on TRIEX, Tactile Automation, and ITRI tools and hardware.

Precision Touch Test Tool (PT3)

You can use either one of these tools for precision touch testing.

- TRIEX/Tactile Precision Touch Test Tool (PT3)
- ITRI Precision Touch Test Tool (PT3)

Rotation and Acoustic (RA) Tool

You can use either one of these tools for RA testing.

- TRIEX/Tactile RA Tool
- ITRI RA Tool

Pen / Stylus Holders and Accessories

Tactile

Part number	Part description	Quantity required
TR10974-006	Stylus Rail Assembly, Weight Configuration (Stylus Holder)	1
TR10975-001	RA Mounting Bracket	1
TR10976-001	PT3 Mounting Bracket	1

Part number	Part description	Quantity required
TR10989-001	Stylus Force Test Tool	1
TR10988-001	Stackable Weight, 1 gram	2 (optional)
TR10988-005	Stackable Weight, 5 gram	3
TR10988-020	Stackable Weight, 20 gram	3
TR10988-100	Stackable Weight, 100 gram	2

ITRI

Part number	Part description	Quantity required
TPL-3-PH-01	Pen Holder for RA Tool	1
HPT-2-PH-01	Pen Holder for PT3	1
LPH-01	Pen Holder for Pressure Test	1
CW-005	Stackable Weight, 25 gram	2
CW-025	Stackable Weight, 50 gram	2
CW-100	Stackable Weight, 100 gram	2

Additional Accessories

In addition to the Pen Holder parts from TRIEX and ITRI, the following accessories are also required for testing.

- Zip Ties (for affixing pen to TRIEX pen holders)

  Recommended part is available from McMaster, part 7130K12

- Ruler/Caliper (for measuring height for Hover Range test)

  Anything that can measure in millimeters will meet requirements

Contacts for Tactile and ITRI

- Tactile Automation, Inc.

  Emery Parks
  +1 (425) 3108380
  emeryp@tactileautomation.com

- Industrial Technology Research Institute

  Chieh-Yu Wu
  +886-3-5743842
  chiehyuwu@itri.org.tw

General Testing Guidelines (general-testing-guidelines)

Article • 03/18/2021

This topic provides general guidelines for testing a Windows pen device, using the Windows Hardware Lab Kit (HLK) for Windows 10.

- Unless otherwise specified, the device should always have AC power when performing HLK tests.
- Unless otherwise specified, all numbers in error messages which report distance or location are in himetric (0.01mm).
- Unless otherwise specified, a nominal weight of 150g should be applied to the pen holder for all tests.
    - When applying the 150g of weight, distribute the weight as evenly as possible onto both sides of the pen holder. This can be either a 50g/100g split, or a 75g/75g split, if enough weights are available.
    - Except for tests that require precise application of weights, it is acceptable that the total applied weight at pen tip exceeds 150g due to the weight of the pen and the pen holder.
- Unless otherwise specified, the pen holder should be held at an angle of 90 degrees i.e. placed vertically against the screen.
- Unless otherwise specified, no input for HLK testing should be collected within 8mm of the edge of the screen.
- Before running each test, ensure the device is lying flat on the PT3 or RA tool.
- Before running the tests, ensure screen resolution is adjusted to the device's default or highest resolution.
- Before running the tests, ensure the device's orientation matches the orientation of the digitizer and the HLK application is full screen.
- The Windows 10 HLK only supports testing a single pen digitizer at a time. If your device exposes multiple pen TLCs, you will need to do the following to ensure that the correct pen digitizer is being tested:
    1. Open **Device Manager**, then expand the **Human Interface Devices** node.

2. Find and disable all HID-compliant pen digitizers, with the exception of the one that you want to validate.

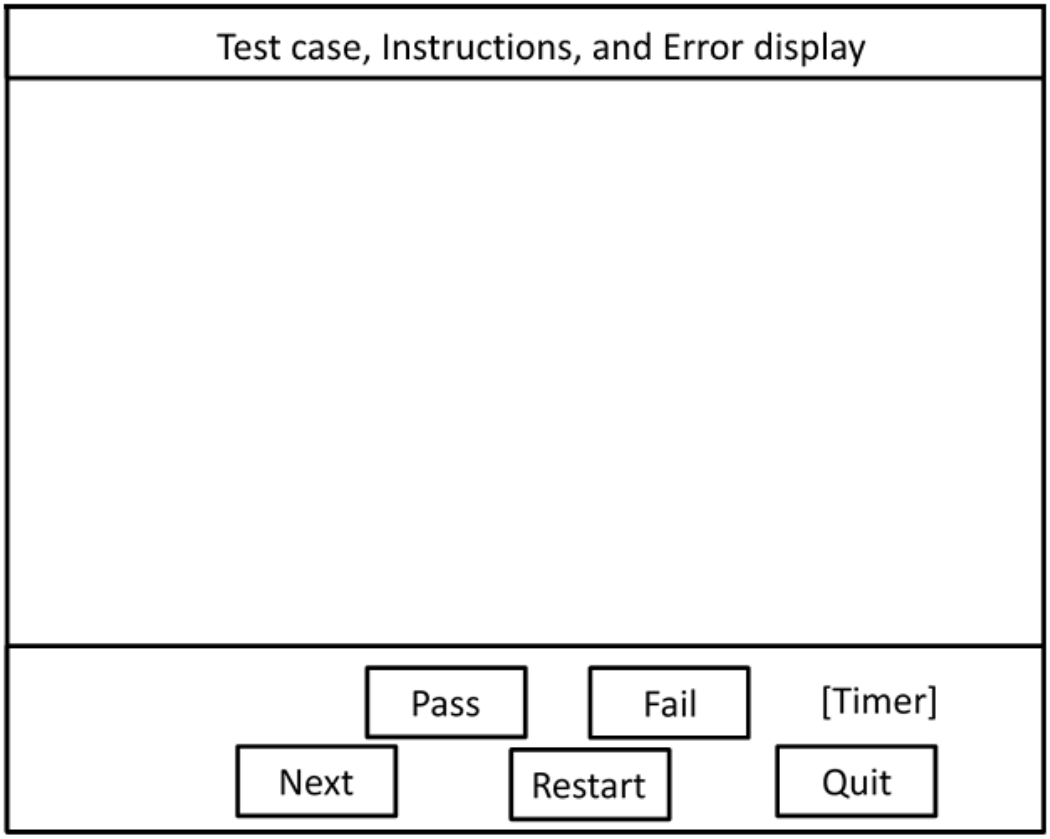
Pen Testing Interface

Article • 01/18/2019

This topic presents the user interface for the pen tests in the Windows Hardware Lab Kit (HLK) for Windows 10.

UI layout

This is the layout of the UI that is used in the pen tests in the HLK.



Buttons available in testing

- Next

  This will cause the current iteration to be skipped and marked as fail and allows the user to proceed to the next iteration.

- Restart

  This will cause this test to be restarted.

- Quit

  This will cause the testing to be ended, remaining tests will be skipped and marked as fail.

- Pass [Manual result only]

  This will cause the current iteration to be marked as pass.

- Fail [Manual result only]

  This will cause the current iteration to be marked as fail.

Active Pen Tests

Article • 05/06/2022

This section lists the topics that provide details about the active pen tests in the Windows Hardware Lab Kit (HLK) for Windows 10.

In this section

Topic	Description
Barrel Button	This is a test for the barrel button feature of a Windows pen device.
Buffering	This is to test the data buffering behavior of a Windows pen device.
Contact accuracy	This is to test the contact position reporting accuracy of a Windows pen device.
Custom Gestures	This is a test to indicate whether or not a Windows pen device supports custom gestures.
Eraser	This is a test for the eraser feature of a Windows pen device.
Ghost Reporting	This is to test the ghost reporting aspect of a Windows pen device.
Hover Accuracy	This is a test of the accuracy of reporting that a Windows pen device is hovering.
Hover Range	This is to test the hovering range for a Windows pen device.
Hover Stationary Jitter	This is to test the jitter aspect of a Windows pen device, when the device is in stationary hover.
Latency - Down	This is a test to check the time lag (latency) between a Windows pen contact with a screen, and when the contact is reported to Windows.
Latency - Move	This is a test to check the time lag (latency) between the Windows pen being in a new screen location, and when that new location is reported to Windows.
Moving Jitter	This is to test the behavior (the jitter aspect) of the Windows pen device, while it's in contact with the screen, and moving.
Physical Dimensions	This is to test the accuracy of the reported screen size (its physical dimensions), by a Windows pen device.

Topic	Description
Pressure Levels	This is to test the accuracy of the reported pressure levels by a Windows pen device.
Pressure Motion	This is to test the consistency of the pressure reporting by a Windows pen device as it moves across a screen.
Resolution	This is to test the reporting resolution of a Windows pen device.
Stationary Jitter	This is to test the position reporting accuracy (jitter) of a Windows pen device that is stationary on the screen.
Third Party Driver	This is to ensure that there is no 3rd party driver for a Windows pen device that is connected via a Windows-supported bus.

Simultaneous pen and touch validation

Article • 06/21/2022

In previous versions of Windows, touch contacts were supressed when pen input was detected within range of the digitizer. Starting in Winodws 1607, devices are able to report simultaneous pen and touch contacts to applications. This topic describes the tools and applications for validating your solution on Windows systems.

Note: On Windows 11 only, before running latency tests, set the Bluetooth & devices > Touch > "Three- and four-finger touch gestures" setting to "Off" as this can introduce a small delay into the touch responsiveness.

Simultaneous pen and touch reporting requirements

Currently, the Windows Hardware Compatibility Program does not contain requirements for simultaneous pen and touch performance. Microsoft instead provides the following recommendations for simultaneous pen and touch performance.

- A device should continue to meet all Windows 10 Pen compatibility requirements when five simultaneous touch contacts are present on the screen.
- A device should continue to meet all Windows 10 touch compatibility requirements when a pen is within reporting range of the screen.

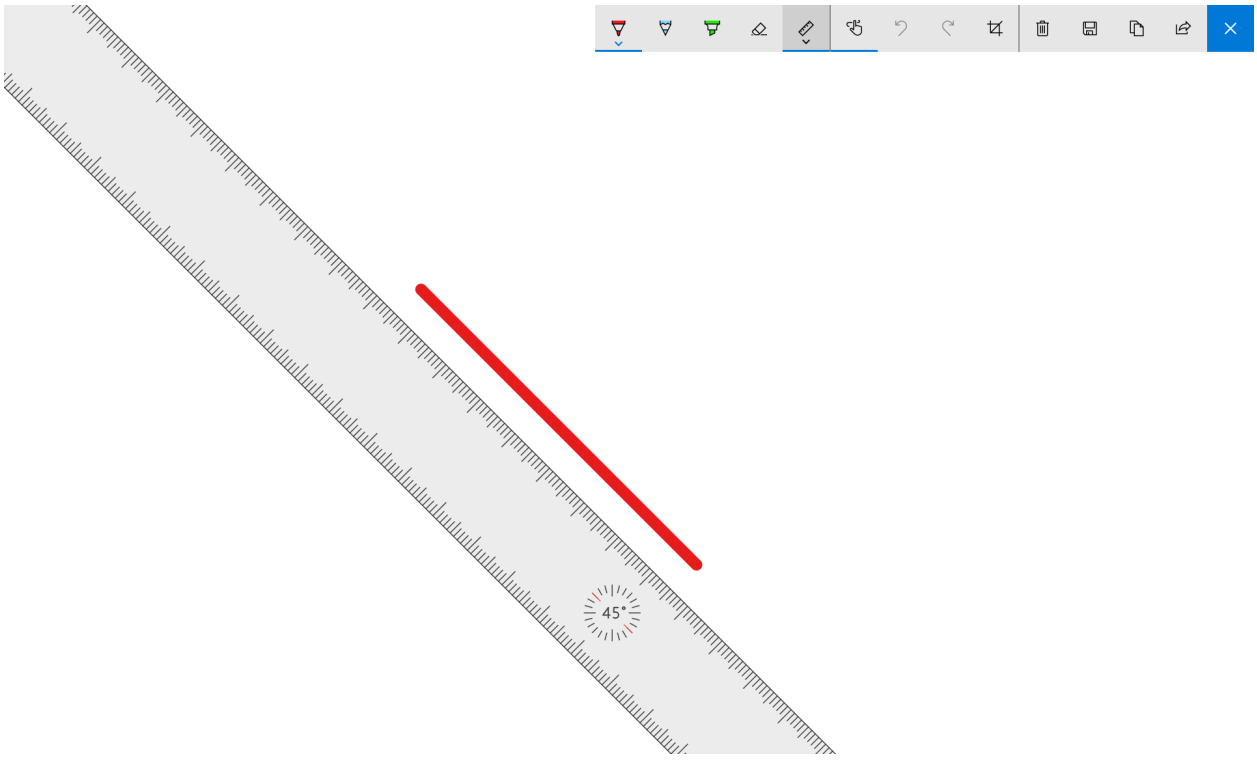
Devices capable of following these recommendations will be able to follow future requirements for pen and touch reporting

Validating Simultaneous Reporting Support

The first step in validating a Simultaneous Pen and Touch (SPT) system is to ensure the system supports sending touch and pen input simultaneously. Windows supports several ways of verifying this using basic applications. The two recommended methods are detailed below.

Windows Ink Workspace

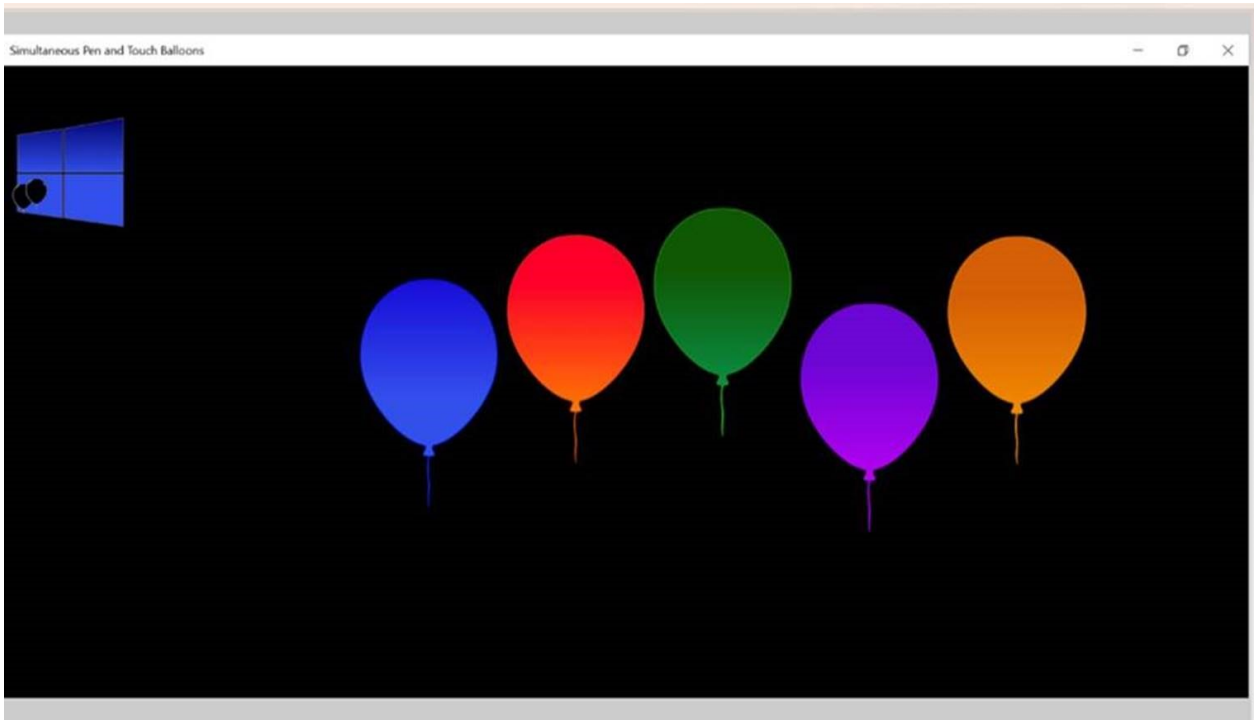
Using the Windows Ink Workspace and a pen device, you can validate an SPT system by using the workspace ruler.



1. Using touch, grab the Ruler and move its location on the screen.
2. Without lifting the touch contact from the screen, bring the pen into contact with the screen on the canvas, then remove the pen contact.
3. If the system supports SPT, you should still be able to move the Ruler with the touch contact that is still against the screen. If the system does not support SPT, you will need to remove the touch contact from the screen and tap the Ruler again to move the Ruler.

Simultaneous pen and touch baloons

You can also use the Simultaneous Pen and Touch Balloons App developed by Blue Lan to determine if a system supports SPT. The app renders several floating balloons on the screen.



To validate SPT on the system, place a finger on a balloon to stop it, and then tap the stopped balloon with your pen. If balloon is popped by the pen, SPT is supported on the system. If the balloon moves or is not popped, SPT is not supported by the touch device's firmware.

Validating pen and touch with the Windows Hardware Lab Kit

To ensure that a device can meet Microsoft's Pen and Touch requirements, Microsoft provides pen and touch tests as part of the Windows Hardware Lab Kit (HLK). Some of these tests can be used to verify that a device meets Microsoft's SPT requirements, as many of the tests now accept both forms of input. The following sections provide detailed guideance on these tests and test proceedures.

> **Note**
>
> Testing simultaneous pen and touch inputs with the HLK requires HLK build 14971 or greater

Pre-testing requirements

In addition to the setup steps required for Touch HLK testing, testing SPT requires you to download and have two additional .reg files:

- Run_Before_SPT_Testing_And_Restart.reg Pen and Touch Validation reg files/Run_Before_SPT_Testing_And_Restart.reg)
- Run_After_SPT_Testing_And_Restart.reg Pen and Touch Validation reg files/Run_After_SPT_Testing_And_Restart.reg)

Be sure to run the appropriate .reg file before SPT testing to enable the correct testing conditions, and after SPT testing to return the system to its normal state.

Validating pen requirements

Ensuring a device meets the SPT pen requirements while touch is present on the system can be performed using the Pen HLK. This validation can be performed by running the Pen HLK tests through HLK Studio or manually.

Windows supports SPT testing on the following Pen HLK tests when performed with five fingers in contact with the screen:

- Buffering
- Edge Gutter
- Resolution
- Moving Jitter
- Hover Accuracy
- Hover Range

For more information on these tests, see the Pen validation guide

Finger placement

To ensure that finger placement does not interfere with the results of the Pen HLK tests, do not place finger contacts in the green and yellow starting/ending sections on a test iteration, in the path of the pen.

For all tests, place all finger contacts against the screen before bringing the pen in range of the screen. Fingers can be moved during the test, but do not lift fingers from the screen until the end of an iteration. Additionally, be careful of bringing touch contacts near the buttons used for controlling test iterations.

Validating touch requirements

Ensuring a device meets the SPT pen requirements while a pen is present on the system can be performed using the Touch HLK. This validation can be performed by running the

Touch HLK tests through HLK Studio or manually.

Windows supports SPT testing on the following Touch HLK tests when performed with the pen in contact with the screen:

- 5 Point Minimum
- Buffering
- Latency - Down
- Reporting Rate

For more information on these tests, see the Touchscreen validation guide

Pen Placement

To ensure the pen placement does not interfere with the results of the Touch HLK tests, place the pen as far from the starting/ending sections, and the path of the fingers as possible. If a finger passes within 12cm of the pen during an iteration, Windows' built in palm rejection may choose to suppress the contact, causing the iteration to fail.

To decrease the likelihood of touch contacts being suppressed, you can verify the system's handedness setting. To do this, open the **Settings** menu, and select **Devices**. On the **Devices** list on the left, select **Pen & Windows Ink**.

If the **Choose which hand you write with** option is set to "Right Hand", placing the pen in the bottom right corner of the screen when a touch target is not in that area will greatly reduce the chance of contacts being suppressed. If it is set to "Left Hand", placing the pen in the bottom left corner of the screen will do the same.

For all tests, bring the pen in contact with the screen before bringing any touch contacts against the screen. The pen can be moved during the test, but do not lift the pen from the screen until the end of an iteration. Be careful when bringing the pen near the buttons used for controlling test iterations, as the pen may interact with them.

Additional testing tools

The test tool Digiinfo can be used to view input from touch and pen devices in real time to inspect issues with input reporting. For SPT validation, Digiinfo can be used simply to validate that input is being received from all five fingers when pen is in range of the screen, that the report rate of each touch contact equals/exceeds 60Hz, and that the report rate of the pen is at least 133Hz.

Digi Info

File Collection View Help

Digitizer Type: Integrated Touch, Integrated Pen, Multi Input, Ready

Sampling Rate: 137 100 100 100 100 100

Server Connection: None

Start Stop Clear Packets

Timestamp	ScanTime	X	Y	Cent...	Cent...	Width	Height	ContactId	ActualCount	Message	Buttons And Flags
360370515	24949	746	756					0	1	WM_INPUT	Confidence Tip
360370515	36300	3066	3146	3066	3146	458	505	132	5	WM_INPUT	Confidence Tip
360370515	36300	2769	5784	2769	5784	611	336	131	5	WM_INPUT	Confidence Tip
360370515	36300	545	3152	545	3152	359	505	130	5	WM_INPUT	

Barrel Button

Article • 09/21/2021

This is a test for the barrel button feature of a Windows pen device.

Test name

- Barrel Button

Core requirements tested

- Device.Input.Digitizer.Pen.BarrelButton

Test purpose

- Verifies that the pen reports barrel button flag correctly.

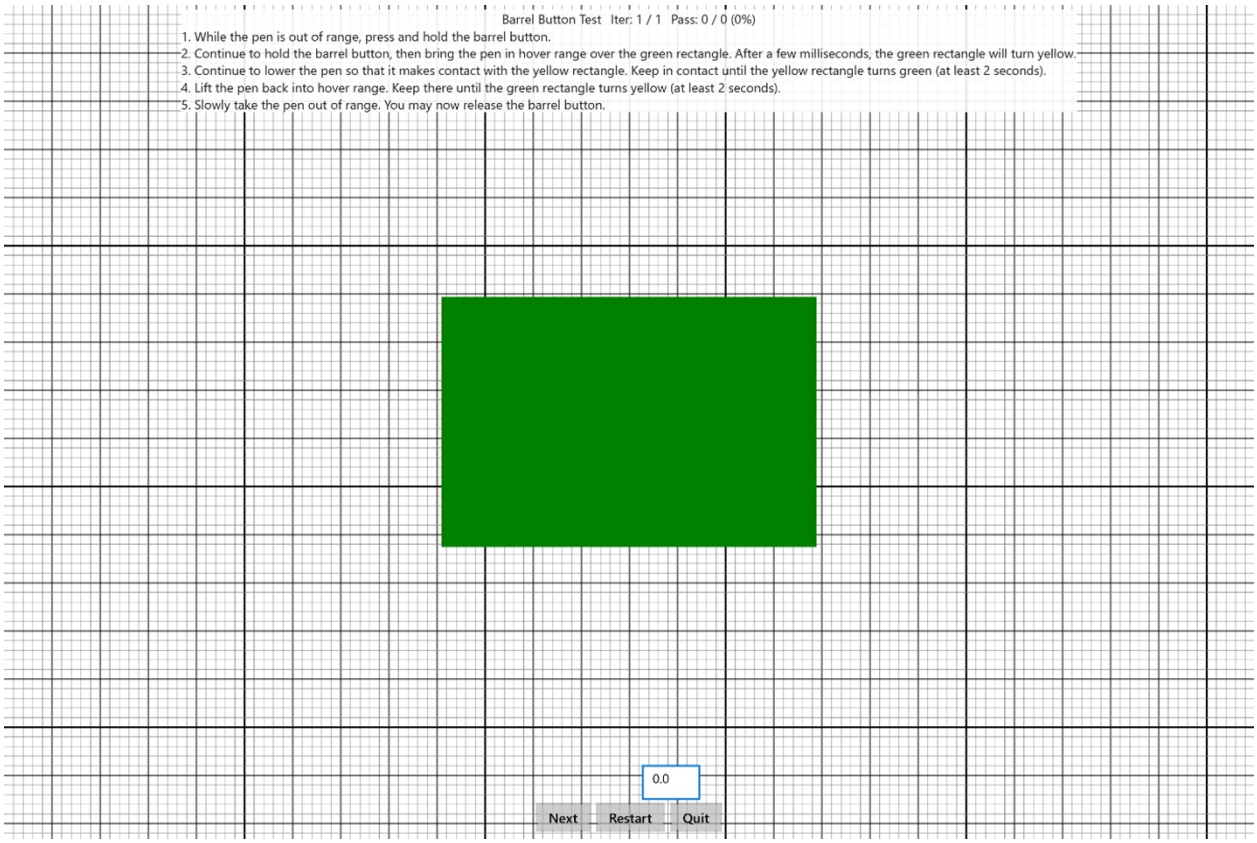
Tools required

- None.

Validation steps

1. While the pen is out of range, press and hold the barrel button.
2. Continue to hold the barrel button, then bring the pen in hover range over the green rectangle. After a few milliseconds, the green rectangle will turn yellow.
3. Continue to lower the pen so that it makes contact with the yellow rectangle. Keep in contact until the yellow rectangle turns green (at least 2 seconds).
4. Lift the pen back into hover range. Keep there until the green rectangle turns yellow (at least 2 seconds).
5. Slowly take the pen out of range. You may now release the barrel button.

Here's a screenshot from the Barrel Button test.


Barrel Button Test Iter: 1 / 1 Pass: 0 / 0 (0%)
1. While the pen is out of range, press and hold the barrel button.
2. Continue to hold the barrel button, then bring the pen in hover range over the green rectangle. After a few milliseconds, the green rectangle will turn yellow.
3. Continue to lower the pen so that it makes contact with the yellow rectangle. Keep in contact until the yellow rectangle turns green (at least 2 seconds).
4. Lift the pen back into hover range. Keep there until the green rectangle turns yellow (at least 2 seconds).
5. Slowly take the pen out of range. You may now release the barrel button.
0.0
Next
Restart
Quit

Common error messages

- "Barrel button went off after being detected"

  Occurs if the barrel button is unexpectedly transitioned from on to off in the middle of the test.

- "Barrel button was not detected in time"

  Occurs if the barrel button was not detected within the allotted time of the pen being placed in hover range.

- "Barrel button was not detected before contact was made"

  Occurs if the pen made contact as instructed in step 3, but the barrel button was never detected.

- "Pen lost contact before rectangle changed color!"

  Occurs when user lifts pen up in step 4 too early, before the rectangle changes back to green.

- "The contact did not begin in the correct location"

  Occurs if the pen stroke does not begin in the green rectangle.

Passing criteria

- 1/1 test iterations must pass for test to succeed.

Buffering (buffering)

Article • 03/18/2021

This is to test the data buffering behavior of a Windows pen device.

Test name

- Buffering

Core requirements tested

- Device.Input.Digitizer.Pen.Buffering

Test purpose

- Verifies that the device has no data loss on wake from low power states.

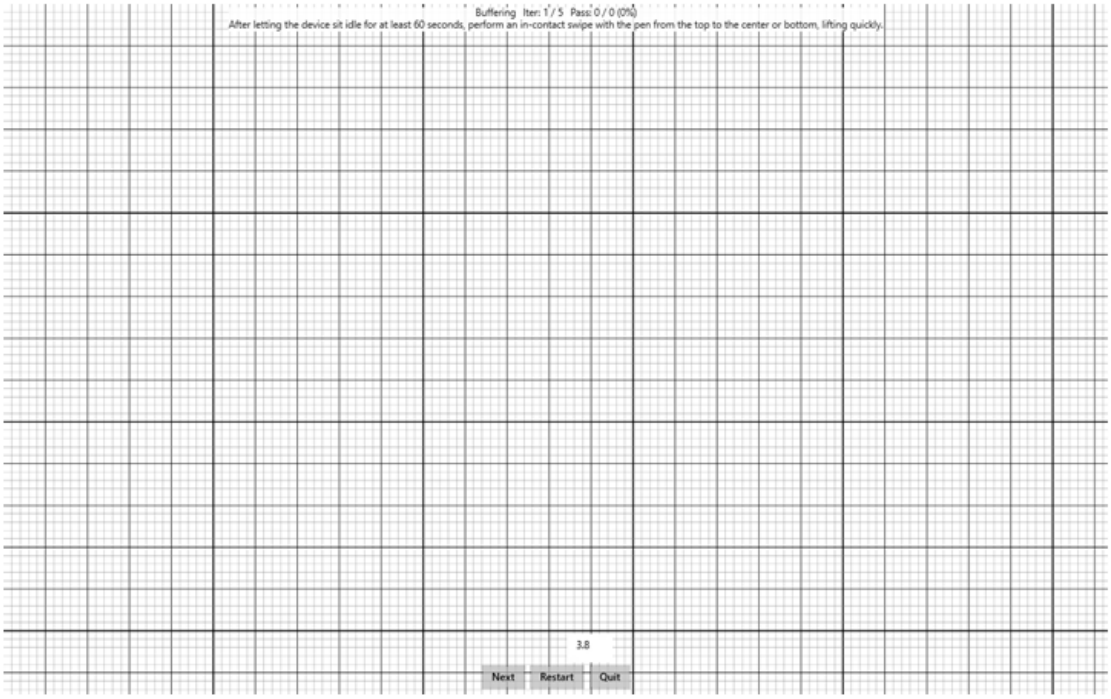
Tools required

- None.

Validation steps

1. Run the test case for buffering.
2. Set the pen out of range of the screen for at least 60 seconds.
3. Following the directions on the screen, you will perform a swipe with the pen across the screen:
   a. The swipe must be quick, the entire interaction (from entering hover range to leaving hover range) must be completed in less than half a second.
   b. The swipe must make contact with the screen.
   c. While in contact with the screen, the pen must make a displacement of > 4mm.

Here's a screenshot from the Buffering test.


Buffering Iter: 1 / 5 Pass: 0 / 0 (0%)
After letting the device sit idle for at least 60 seconds, perform an in-contact swipe with the pen from the top to the center or bottom, lifting quickly.
3.8
Next
Restart
Quit

Common error messages

- "Pen was not idle long enough"

  Occurs if the device was not left in idle state for at least 60 seconds.

- "Stroke displacement too low"

  Occurs if the pen swipe did not make a displacement of > 4mm.

- "The interaction's duration was too long"

  Occurs if the pen swipe lasted more than half a second.

Passing criteria

- 1/1 test iterations must pass for test to succeed.

Contact accuracy

Article • 01/18/2019

This is to test the contact position reporting accuracy of a Windows pen device.

Test name

- ContactAccuracy

Core requirements tested

- Device.Input.Digitizer.Pen.Accuracy

Test purpose

- Verifies that while the pen is in contact with the screen, the physical contact with the device matches the contact position that the device reports - within allowed limits.

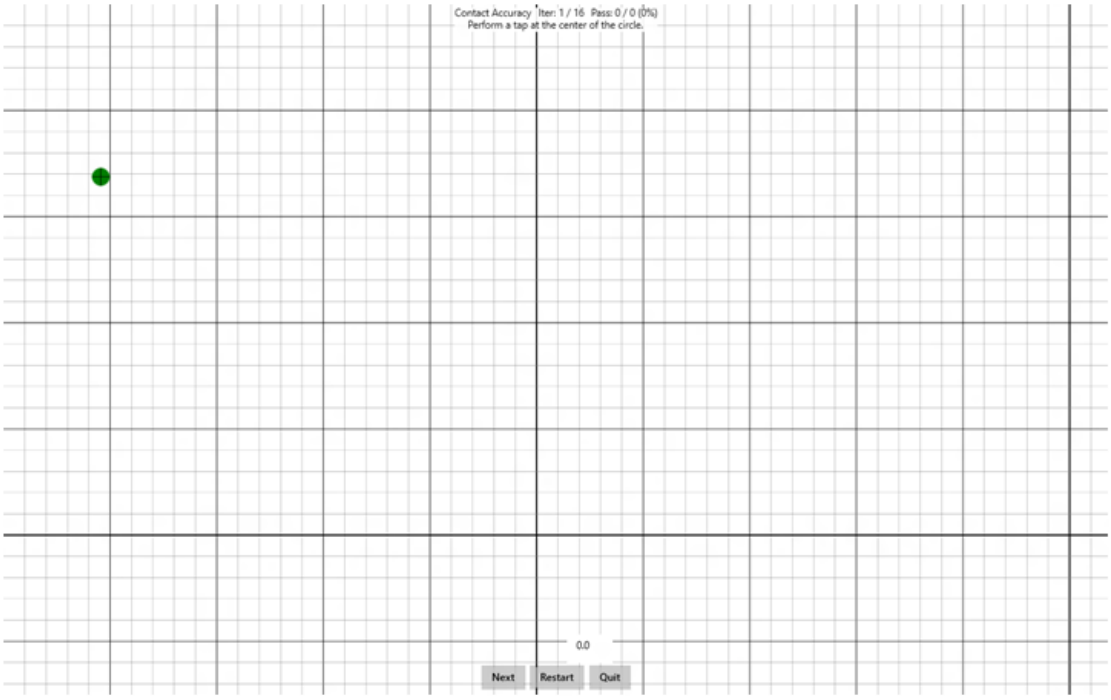
Tools required

- PT3 device with Pen holder + 150g weight.

Validation steps

1. Run the test case for Contact Accuracy.
2. Following the directions on the screen, tap at the center of the circle located at the cross hair. For each iteration:
   a. Tap the center of the location specified on the screen.

Here's a screenshot from the Contact Accuracy test.



Common error messages

- "Contact distance from target too large:[hm]"

  Occurs if the tap is too far away from the center of the target. Distance given in hundredths of a millimeter.

Passing criteria

- 14/16 test iterations must pass for test success.
- This test has an allowed tolerance of 0.1mm, therefore a recorded contact accuracy <= 0.6 mm will result in a pass.
- Any iteration that is 0.1mm beyond accepted tolerance, resulting in a recorded contact accuracy >= 0.7 mm, will fail the entire test.

Custom Gestures

Article • 01/18/2019

This is a test to indicate whether or not a Windows pen device supports custom gestures.

Test name

- Custom Gestures

Core requirements tested

- Device.Input.Digitizer.Base.CustomGestures

Test purpose

- Verifies that the pen does not have custom runtime gestures.

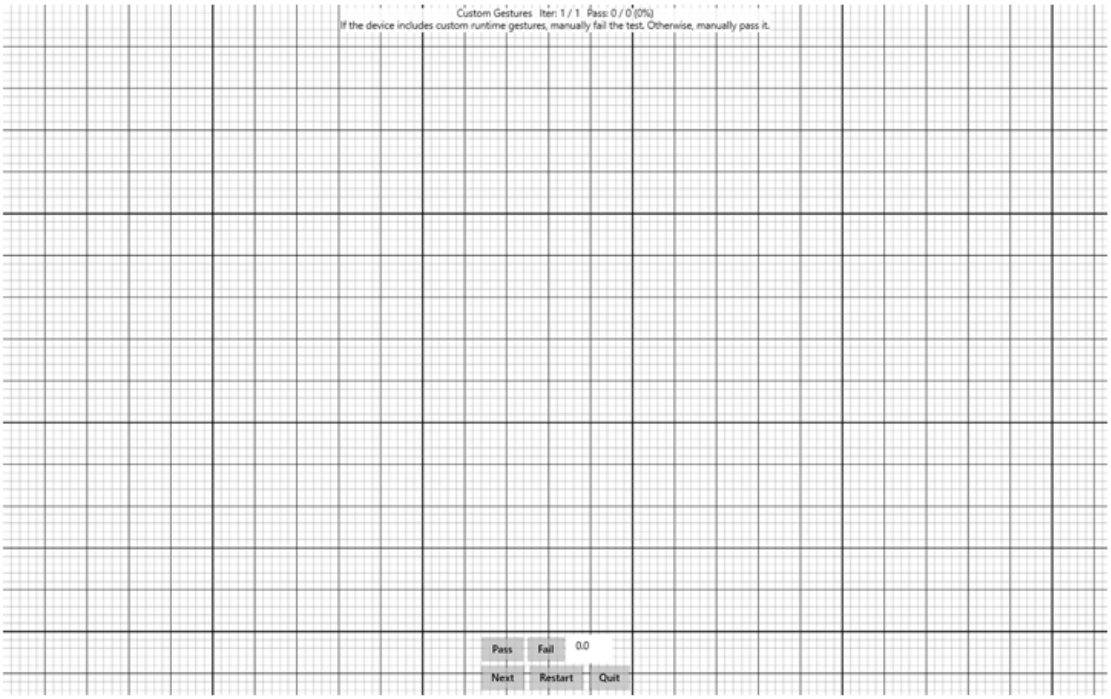
Tools required

- None.

Validation steps

1. Run test case for Custom Gestures.
2. Click the "Fail" button, if your device has custom runtime gestures, otherwise click "Pass."

Here's a screenshot from the Custom Gestures test.


Custom Gestures Iter: 1 / 1 Pass: 0 / 0 (0%)
If the device includes custom runtime gestures, manually fail the test. Otherwise, manually pass it.
Pass
Fail
0.0
Next
Restart
Quit

Common error messages

- "Manually failed by user"

  Occurs if the operator manually Fails the test (fails the Custom Gestures requirement).

Passing criteria

- 1/1 test iterations must pass for the test to succeed.

Eraser

Article • 01/18/2019

This is a test for the eraser feature of a Windows pen device.

Test name

- Eraser

Core requirements tested

- Device.Input.Digitizer.Pen.Eraser.

Test purpose

- Verifies that the pen reports eraser button flag correctly.

Tools required

- None (PT3 with Pen Holder optional).

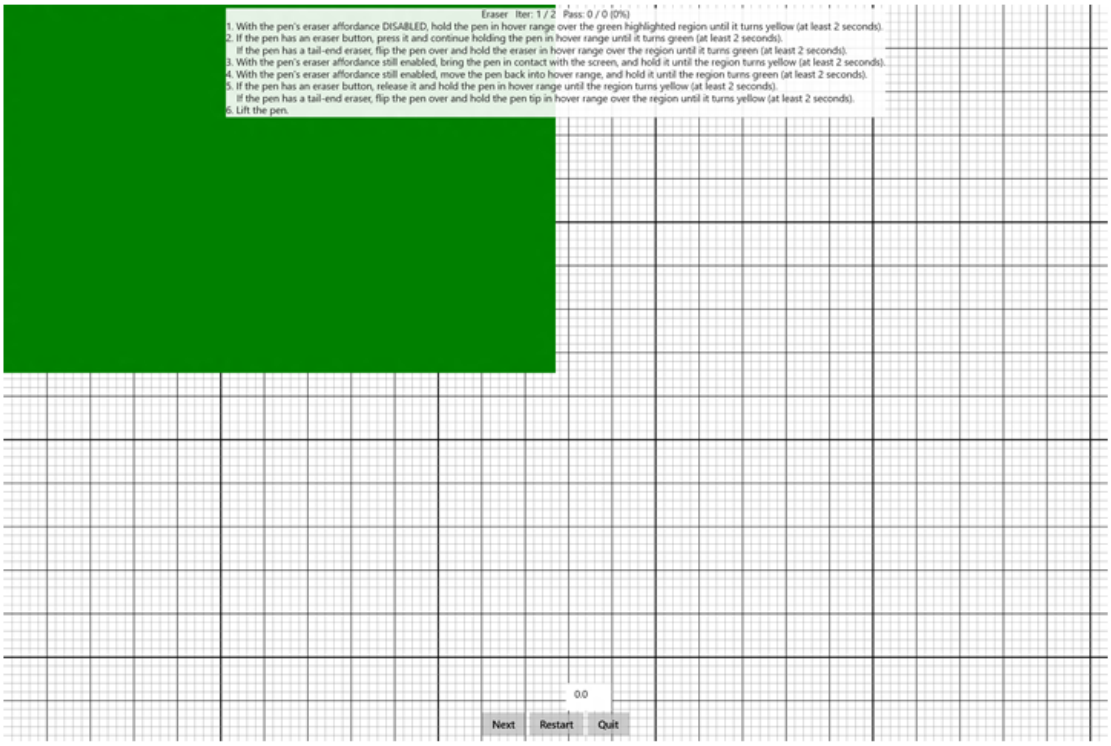
Validation steps

1. Run the test case for Eraser.
2. With the pen's eraser affordance disabled, hold the pen in hover range over the green highlighted region until it turns yellow (at least 2 seconds).
3. Enable the pen’s eraser affordance and hold the pen in hover range until the yellow highlighted region turns green (at least 2 seconds).
    - If the pen has an eraser button, press it and continue holding the pen in hover range.
    - If the pen has a tail-end eraser, flip the pen over and hold the eraser in hover range over the region.
4. With the pen's eraser affordance still enabled, bring the pen in contact with the screen, and hold it until the green highlighted region turns yellow (at least 2 seconds).
5. With the pen's eraser affordance still enabled, move the pen back into hover range, and hold it until the yellow highlighted region turns green (at least 2 seconds).
6. Disable the pen’s eraser affordance and hold the pen in hover range until the green highlighted region turns yellow (at least two seconds).

- If the pen has an eraser button, release it and continue holding the pen in hover range.
- If the pen has a tail-end eraser, flip the pen over and hold the pen tip in hover range over the region.

7. Lift the pen out of hover range.

Here's a screenshot from the Eraser test.


Eraser Iter: 1 / 2 Pass: 0 / 0 (0%)
1. With the pen's eraser affordance DISABLED, hold the pen in hover range over the green highlighted region until it turns yellow (at least 2 seconds).
2. If the pen has an eraser button, press it and continue holding the pen in hover range until it turns green (at least 2 seconds).
If the pen has a tail-end eraser, flip the pen over and hold the eraser in hover range over the region until it turns green (at least 2 seconds).
3. With the pen's eraser affordance still enabled, bring the pen in contact with the screen, and hold it until the region turns yellow (at least 2 seconds).
4. With the pen's eraser affordance still enabled, move the pen back into hover range, and hold it until the region turns green (at least 2 seconds).
5. If the pen has an eraser button, release it and hold the pen in hover range until the region turns yellow (at least 2 seconds).
If the pen has a tail-end eraser, flip the pen over and hold the pen tip in hover range over the region until it turns yellow (at least 2 seconds).
6. Lift the pen.
0.0
Next
Restart
Quit

Common error messages

- "Button flags set incorrectly"

  Occurs if any packet contains incorrect button flags relative to the pen's current state.

- "Pen was not hovering or in contact for long enough"

  Occurs if the pen is not left hovering or in contact for the required duration.

- "The contact did not begin/end in the correct location"

  Occurs if the pen stroke does not begin and end in the green rectangle.

Passing criteria

- 2/2 test iterations must pass for the test to succeed.

Ghost Reporting (ghost-reporting)

Article • 03/18/2021

This is to test the ghost reporting aspect of a Windows pen device.

Test name

- Pen.GhostReporting

Core requirements tested

- Device.Input.Digitizer.Base.ContactReports

Test purpose

- Verifies that the pen does not report ghost points.

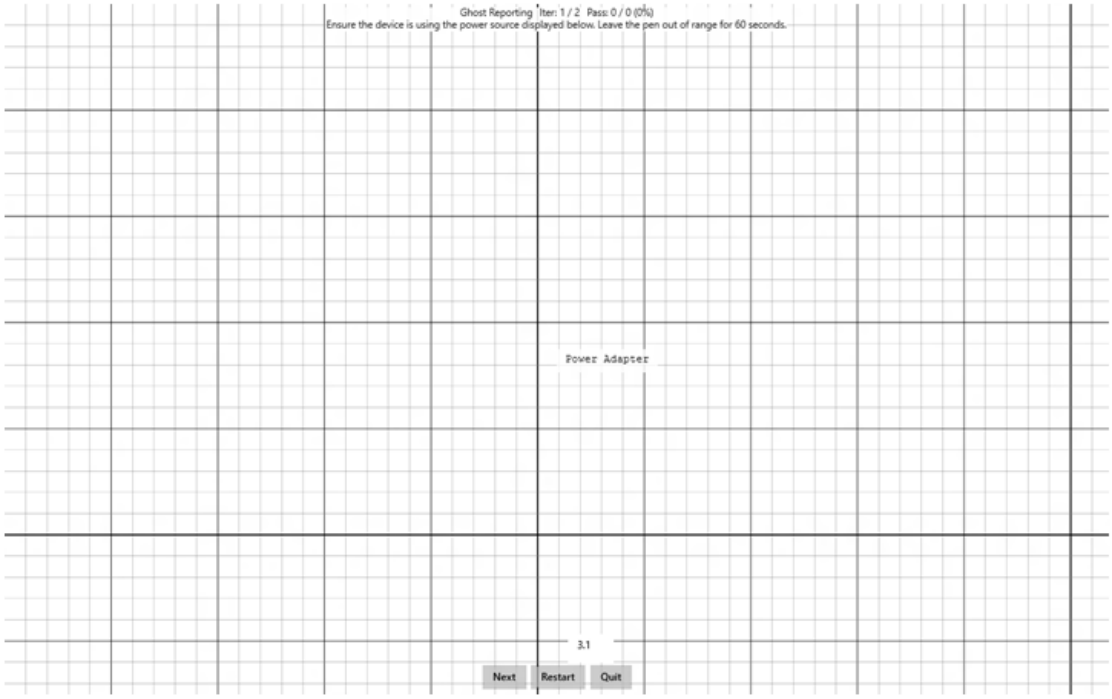
Tools required

- None.

Validation steps

1. Ensure that the device is using the power source.
2. Leave the pen out of range for 60 seconds.
3. Disconnect power source.
4. Leave the pen out of range for 60 seconds.

Here's a screenshot from the Ghost Reporting test.


Ghost Reporting Iter: 1 / 2 Pass: 0 / 0 (0%)
Ensure the device is using the power source displayed below. Leave the pen out of range for 60 seconds.
Power Adapter
3.1
Next
Restart
Quit

Common error messages

- "Ghost input detected"

  Occurs if any pen input is detected during the test.

Passing criteria

- 2/2 test iterations must pass for the test to succeed.

Hover Accuracy

Article • 01/18/2019

This is a test of the accuracy of reporting that a Windows pen device is hovering.

Test name

- Pen.HoverAccuracy

Core requirements tested

- Device.Input.Digitizer.Pen.Accuracy – Hover Accuracy.

Test purpose

- Verifies that the pen reports hover accuracy within prescribed limits.

Tools required

- PT3 with Pen holder.

Validation steps

1. Slowly lower the pen down to approximately 5 mm from the screen (in hover range) in the highlighted quadrant.
2. Once the quadrant turns yellow, after a couple of seconds, lower the pen as slowly as possible until it hits the screen.
3. Once the pen is in contact with the screen, the color turns green.
4. Then start moving the pen away from the screen upwards.

Here's a screenshot from the Hover Accuracy test.

Hover Accuracy Iter: 1 / 8 Pass: 0 / 0 (0%)
Lower the pen in the highlighted quadrant to the minimum hover height. Wait for the region to turn yellow (5 seconds after the pen is detected), then slowly lower the pen until it hits the screen. When the region turns green again, lift the pen.
0.0
Next
Restart
Quit

Common error messages

- "The contact did not begin/end in the correct location"

  Occurs if the pen stroke does not begin and end in the green rectangle.

- "Input did not make contact with the screen"

  Occurs if the pen interaction never touches the screen.

- "Contact distance from hover point too large”

  Occurs if the pen’s hover accuracy does not pass the requirement.

Passing criteria

- 7/8 test iterations must pass for the test to succeed.
- This test has a 0.2 mm allowed tolerance, therefore a recorded hover accuracy <= 1.2 mm will result in a pass.
- Any iteration that is 0.2 mm beyond the allowed tolerance, resulting in a recorded hover accuracy >= 1.4 mm, will fail the entire test.

Hover Range

Article • 01/18/2019

This is to test the hovering range for a Windows pen device.

Test name

- Pen.HoverRange

Core requirements tested

- Device.Input.Digitizer.Pen.HoverRange

Test purpose

- Verifies that the pen has sufficient hover range.

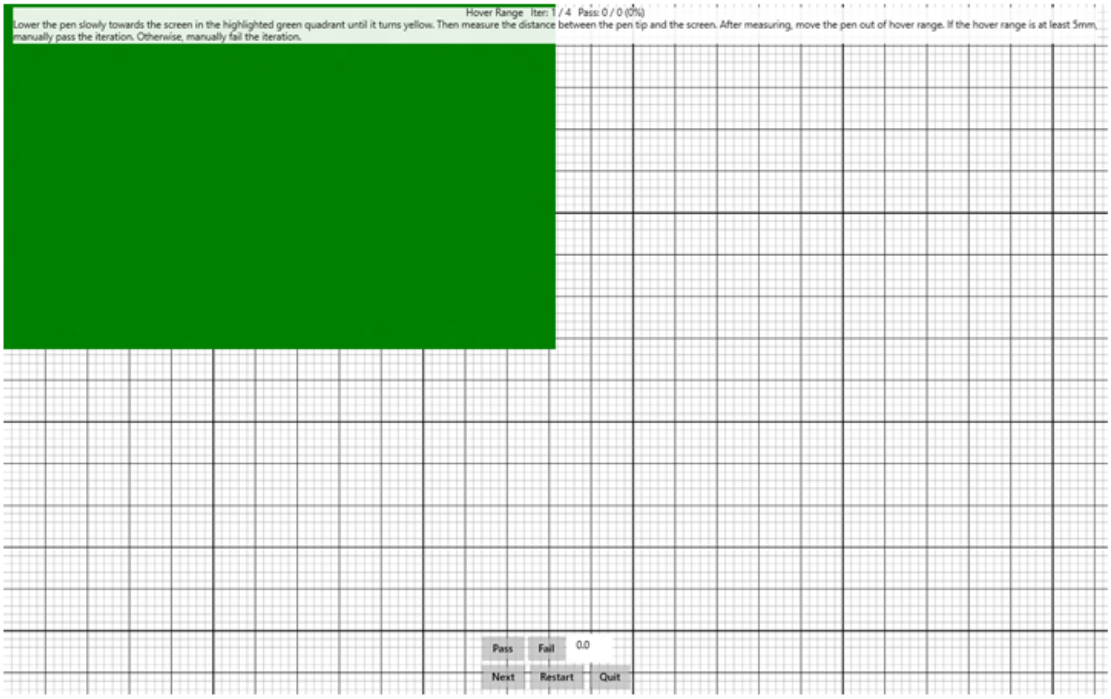
Tools required

- PT3 tool with pen holder.
- Ruler or caliper.

Validation steps

1. Lower the pen slowly towards the screen in the highlighted quadrant.
2. The color of the quadrant turns yellow once (x,y) are reported.
3. Measure the distance between the pen tip and the screen by using a ruler or caliper.
4. After measuring, move the pen out of hover range.
5. If the distance measured was equal to or more than 5 mm, hit Pass. Else hit Fail button.

Here's a screenshot of the Hover - Range test.


Hover Range Iter: 1 / 4 Pass: 0 / 0 (0%)
Lower the pen slowly towards the screen in the highlighted green quadrant until it turns yellow. Then measure the distance between the pen tip and the screen. After measuring, move the pen out of hover range. If the hover range is at least 5mm, manually pass the iteration. Otherwise, manually fail the iteration.
Pass
Fail
0.0
Next
Restart
Quit

Common error messages

- "Contact is not hovering in the correct quadrant"

  Occurs if pen input is detected in an area other than the highlighted quadrant.

- "Manually failed by user"

  Occurs if the operator manually Fails the test (Fails the hover range requirement).

Passing criteria

- 4/4 test iterations must pass for the test to succeed.

Hover Stationary Jitter

Article • 01/18/2019

This is to test the jitter aspect of a Windows pen device, when the device is in stationary hover.

Test name

- HoverStationaryJitter

Core requirements tested

- Device.Input.Digitizer.Pen.Jitter - Hover Stationary jitter.

Test purpose

- Verifies that the jitter that is detected when a pen is doing a stationary hover, occurs within prescribed limits.

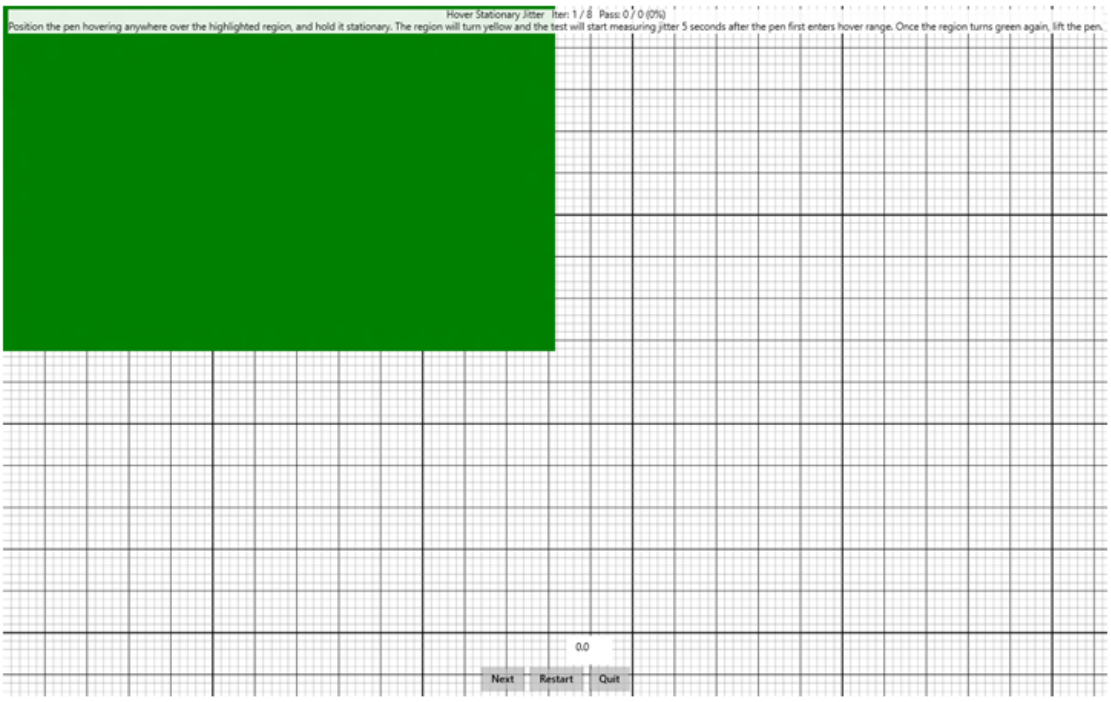
Tools required

- PT3 with Pen holder.

Validation steps

1. Position the pen anywhere over the highlighted region with hover height at approximately 5 mm.
2. Wait a couple of seconds for the highlighted region to turn yellow, which will start recording the pen data.
3. Once the highlighted region turns green again, the data recording is complete.
4. Slowly move the pen away from hover range.

Here's a screenshot from the Hover Stationary Jitter test.


Hover Stationary Jitter Iter: 1 / 8 Pass: 0 / 0 (0%)
Position the pen hovering anywhere over the highlighted region, and hold it stationary. The region will turn yellow and the test will start measuring jitter 5 seconds after the pen first enters hover range. Once the region turns green again, lift the pen.
0.0
Next
Restart
Quit

Common error messages

- "The contact did not begin/end in the correct location."

  Occurs if the pen stroke does not begin and end in the green rectangle.

- "Input made contact with the screen."

  Occurs if the pen interaction touches the screen.

- "Input ended too early."

  Occurs if the pen is not held stationary in hover range for the required duration.

- "Jitter too high."

  Occurs if the pen does not pass the hover stationary jitter requirement.

Passing criteria

- 7/8 test iterations must pass for test to succeed.
- This test has a 0.10 mm allowed tolerance, therefore a recorded hover stationary jitter of <= 0.6 mm will result in a pass.
- Any iteration that is 0.15mm beyond the allowed tolerance, resulting in a recorded hover stationary jitter of >= 0.7 mm, will fail the entire test.

- Recorded value for Hover Stationary jitter is calculated as maximum hover jitter over the iteration.

Latency - Down

Article • 11/05/2020

This is a test to check the time lag (latency) between a Windows pen contact with a screen, and when the contact is reported to Windows.

Test name

- Down Latency

Core requirements tested

- Device.Input.Digitizer.Pen.Latency – Down Latency

Test purpose

- Verifies that when the pen makes initial contact with the screen, the lag between the pen being at a location, and having the location reported to Windows is within allowed limits.

Tools required

- RA Tool

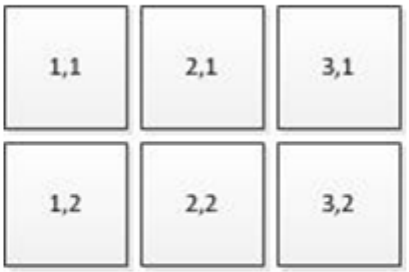
Validation steps

1. For detailed instructions for this test, see How to measure Touch Down Latency by using an Acoustic Measurement Tool.
   a. Note: when directed to contact screen with finger, use pen instead.
   b. For each tap, ensure that the pen starts within hover range (approximately 5mm above the screen), and then contacts the screen with enough force to exceed the pen's activation force. And also make sure that the contact point is more than 8mm from any edge of the screen.
   c. Only 50 taps during the collection phase are required for this class of device.

2. When performing the test, virtually divide the screen into six random regions to prepare to uniformly cover all six areas. The difference of sample counts in any two regions cannot be more than 10%; that is, if you tap 50 times for all regions, you cannot tap more than 9 times for any single region. This ensures that the latency of each region is equally represented in the final average latency.
   a. If you plan to move the microphone as you cover each region, place the microphone in the first area that you plan to tap.
   b. Alternatively, you can place the microphone near the center area (for example, between (**2,1**) and (**2,2**) in the following diagram). In this case, you must move

the microphone at least one time to also cover the original placement of the microphone.

Here's a screenshot from the Down Latency test.



Common errors

- During the collection phase, any palm or touch contact with the device before the pen contacts the screen, could be picked up by the microphone. Please avoid contacting the screen with anything other than the pen, and refrain from bumping the RA tool, or making loud noises during the collection phase.

Passing criteria

- 50 iterations between all screen regions must be collected, with a passing average pen down latency for test success.
- This test has a 7ms allowed tolerance, therefore an average pen down latency of <=42ms will result in a pass.

Latency - Move

Article • 01/18/2019

This is a test to check the time lag (latency) between the Windows pen being in a new screen location, and when that new location is reported to Windows.

Test name

- Move Latency

Core requirements tested

- Device.Input.Digitizer.Pen.Latency – Move Latency

Test purpose

- Verifies that while the pen is in contact with the screen, the lag between the pen being at a location and having the location reported to Windows, is within allowed limits.

Tools required

- RA Tool with Pen holder + 150g weight.
- Zip ties are needed, if you purchase Triex pen test kit.

Validation steps

1. Make sure that the RA tool is fitted with the Pen Holder, and attach the pen to the pen holder with zip ties. Ensure that none of the barrel buttons are activated when the pen is in the pen holder. The Triex RA tool with the pen attached to the pen holder, is shown in the following image:

2. Plug the RA tool control box into the test machine, *and then* turn on the control box. The control box might end up in a weird state if this is done in a different order. Set the switches on the control box as follows:
   a. Motor: Off
   b. Calibrate: Off
   c. Mode: Tachometer

3. Adjust the radius of the circle to approximately 23-25 mm and position the device, so that the contact will stay on the digitizer surface during its full rotation. The pen should be placed close to the center of the screen.

    - To make it simpler to verify the radius of the circle, open MSPaint on the device, bring the pen into contact with the screen, and perform a full rotation of the pen. You can measure the circle drawn by the pen with a ruler to determine the radius.

4. Rotate the arm until the notch on the disk is 2-3mm past the photo-interrupter.

    - If it is too close to the photo-interrupter, you will get an extra data point at the beginning of the test. This is because when started, the motor will jerk backwards. If the notch falls under the photo-interrupter during this, it will cause erroneous data.
    - If it is too far from the photo-interrupter, count will seem to increment by 2, instead of giving 2 increments of 1. The motor is calibrated to rotate around and pass slowly for a certain segment of the circle relative to where it started,

and that slow section is expected to pass under the photo-interrupter. But if it starts too far away from the photo-interrupter, the arm will still be moving too quickly when the notch passes under the photo-interrupter.

5. Run test for pen move latency.
6. Lower the contact onto the surface.
7. Press the Spacebar on the test machine and flip the Motor switch to on to begin the Calibration phase.
8. After 9 rotations (when Count on the screen equals 17), the motor will come to a complete stop for 5 seconds. Press the Spacebar during those 5 seconds to advance to Acquisition phase.
9. Wait for 10 rotations of the arm, then press the Spacebar to move onto the Collection phase.
10. After the arm has made 10 rotations in the Collection phase, press the Spacebar one last time to close the test, and then switch off the motor.

Note Unless there's an error, do not exit the test by pressing ESC, otherwise the test might not finish running.

Common errors

- At the beginning of the Calibration stage, if the slot on the disk is too close to the photo interrupt signal, the calibration text will appear with a count of zero (0) before the arm has rotated once. To avoid this issue, do the following:
  - Be sure to position the slot 2-3mm past the photo interrupt signal.
- When calibration executes correctly, the count should increment by 2 every rotation: once before, and once after hitting the photo interrupt signal.
  - Because of this, the pause before the beginning of the Acquisition stage always occurs when the count = 17.
  - If the count isn't incrementing properly, abort the test. Turn off the control box, unplug and re-plug the control box into the test machine via USB, then turn the control box on again. Start the test from the beginning, and verify that the count is incrementing correctly.

  Note A correctly incrementing count should start at 1, and increment by 2 each rotation cycle until the count hits 17.
- At the end of the Calibration stage, the motor will pause completely for 5 seconds. During that pause, the space bar must be hit to advance to the Acquisition stage,

otherwise an error message will be displayed.

Passing criteria

- 10 rotation cycles must be collected with a passing average pen move latency for test success.
- This test has a 6ms allowed tolerance, therefore an average pen move latency of <= 36ms will result in a pass.

Moving Jitter

Article • 01/18/2019

This is to test the behavior (the jitter aspect) of the Windows pen device, while it's in contact with the screen, and moving.

Test name

- MovingJitter

Core requirements tested

- Device.Input.Digitizer.Pen.Jitter – Moving Jitter.
- Device.Input.Digitizer.Pen.ReportRate.

Test purpose

- Verifies that while pen is in motion, and in contact with the screen, the maximum orthogonal distance between points from line of best fit, is within allowed limits.

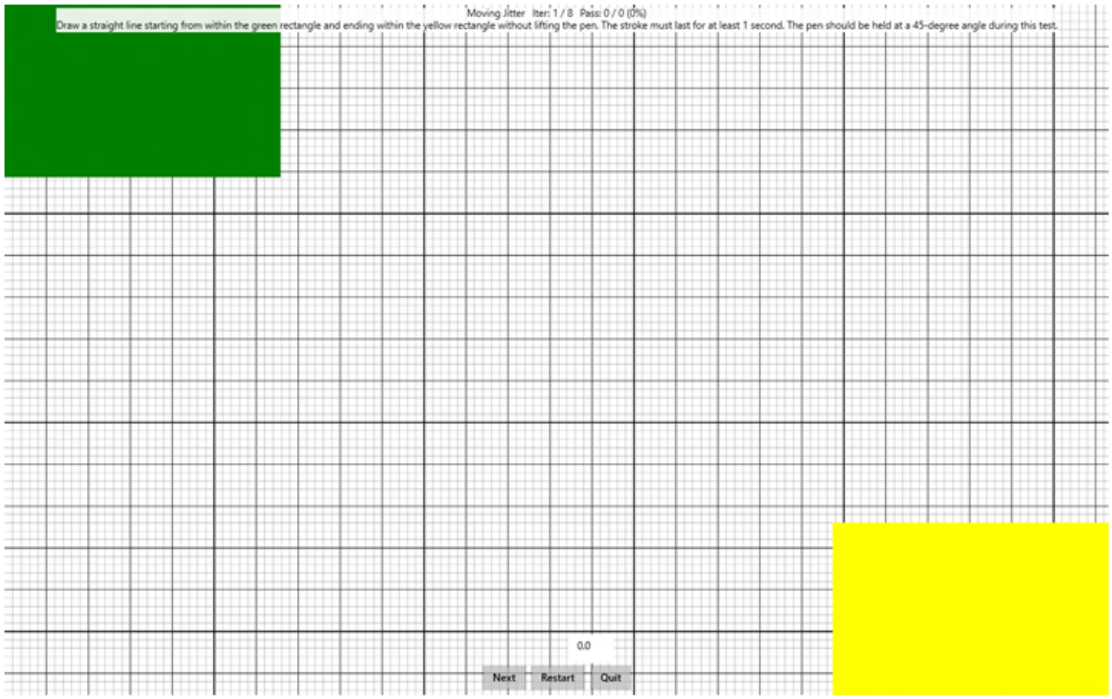
Tools required

- PT3 with Pen holder + 150g weight.

Validation steps

1. Run the test case for Moving jitter.
2. For the purpose of this test, the pen should be held at an angle of 45 degrees in the pen holder.
3. Following the instructions on the screen, draw a line starting from within the green rectangle, and ending within the yellow rectangle without lifting the pen. The stroke must last for at least 1 second.

Here's a screenshot from the Moving Jitter test.


Moving Jitter Iter: 1 / 8 Pass: 0 / 0 (0%)
Draw a straight line starting from within the green rectangle and ending within the yellow rectangle without lifting the pen. The stroke must last for at least 1 second. The pen should be held at a 45-degree angle during this test.
0.0
Next
Restart
Quit

Common error messages

- "Distance from line of best fit too large."

  Occurs if the pen stroke's jitter deviates too far from the line of best fit (defined as the straight line between the first and last packets).

- "The contact did not begin/end in the correct location."

  Occurs if the pen stroke does not begin in the green rectangle and end in the yellow rectangle.

- "The interaction's duration was too short."

  Occurs if the pen stroke does not last the required duration.

- "Report rate too low."

  Occurs if the pen fails the report rate requirement of >= 133 Hz with a tolerance of 7 Hz.

Passing criteria

- 7/8 test iterations must pass for test success.
- This test has an allowed tolerance of 0.05 mm for moving jitter, therefore a recorded moving jitter <= 0.45 mm will result in the iteration passing.

- Any iteration that is more than 0.1 mm beyond the allowed tolerance, resulting in a recorded moving jitter >= 0.5 mm, will fail the entire test.

Physical Dimensions

Article • 01/18/2019

This is to test the accuracy of the reported screen size (its physical dimensions), by a Windows pen device.

Test name

- Physical Dimensions

Core requirements tested

- Device.Input.Digitizer.Pen.HIDCompliance

Test purpose

- Verifies that the size of the screen reported by the device's digitizer, matches its physical size.

Tools required

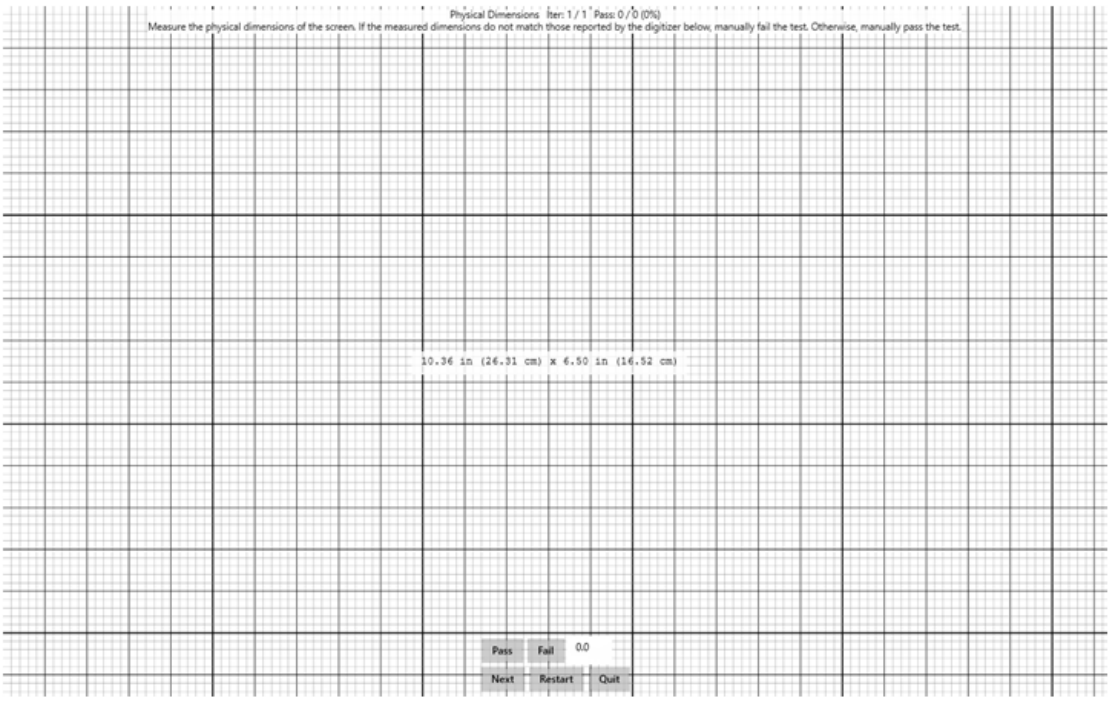
- Ruler.

Validation steps

1. Run the test case for Physical Dimensions.
2. Using a ruler, measure the physical dimensions (width and height) of the device's screen. If the dimensions measured do not match the values reported in the text box in the center of the screen, manually Fail the test. Otherwise, manually Pass the test.

   Note Physical measurements should be accurate to +/- 2mm.

Here's a screenshot from the Physical Dimensions test.


Physical Dimensions Iter: 1 / 1 Pass: 0 / 0 (0%)
Measure the physical dimensions of the screen. If the measured dimensions do not match those reported by the digitizer below, manually fail the test. Otherwise, manually pass the test.
10.36 in (26.31 cm) x 6.50 in (16.52 cm)
Pass
Fail
0.0
Next
Restart
Quit

Common error messages

- None.

Passing criteria

- 1/1 test iterations must pass for test to succeed.

Pressure Levels

Article • 01/18/2019

This is to test the accuracy of the reported pressure levels by a Windows pen device.

Test name

- PressureLevels

Core requirements tested

- Device.Input.Digitizer.Pen.Pressure.

Test purpose

- Verifies that the pen has a logarithmic reported pressure value between activation force and 250 g of input applied force.

Tools required

- Pressure Jig (TRIEX Stylus Force Test Tool or ITRI Pen Holder for Pressure Test) and weights.
- Zip ties are needed, if you decide to use Triex pen test kit.

Validation steps

1. Run the test case for Pressure Levels.

2. Note that this test requires different weights to be applied to the pen holder instead of the nominal weight.

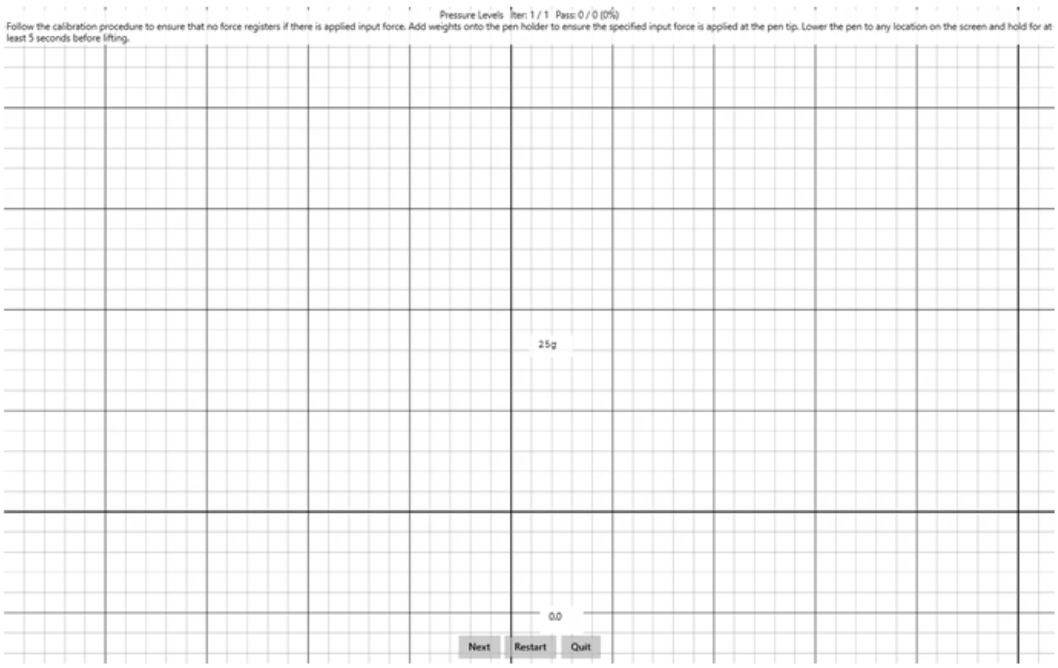
3. Follow the calibration procedure for the pen on the pressure jig.
   a. Attach the pen to the pen holder of the pressure jig.
   b. Adjust the counterweights so that the pen is perfectly balanced with 0 grams of pressure.
   c. Do not re-calibrate the pressure jig in the middle of the test. If the jig requires additional calibration after you've started the test, re-calibrate the jig and restart the test.

4. Add weights onto the pen holder on the pressure jig to match the weight being displayed on the screen.
   a. The weights shown on the screen will be 7g, 15g, 25g, 50g, 75g, 100g, 125g, 150g, 175g, 200g, and 250g.

b. The 7g and 15g iterations are optional to provide better accuracy for devices that exceed the activation force requirement – there is no penalty for skipping these iterations.

5. Slowly and softly lower the pen to any location on the screen, and keep the pen in contact with the screen for at least 5 seconds.
6. After 5 seconds have passed, lift the pen from the screen from the bottom of the pen holder, to ensure additional pressure is not recorded by the test.

Here's a screenshot from the Pressure Levels test.


Pressure Levels Iter: 1 / 1 Pass: 0 / 0 (0%)
Follow the calibration procedure to ensure that no force registers if there is applied input force. Add weights onto the pen holder to ensure the specified input force is applied at the pen tip. Lower the pen to any location on the screen and hold for at least 5 seconds before lifting.
2.5g
0.0
Next
Restart
Quit

Common error messages

- "Reported pressure levels exceed tolerance."

  Occurs if the reported pressure levels are too far from logarithmic curve.

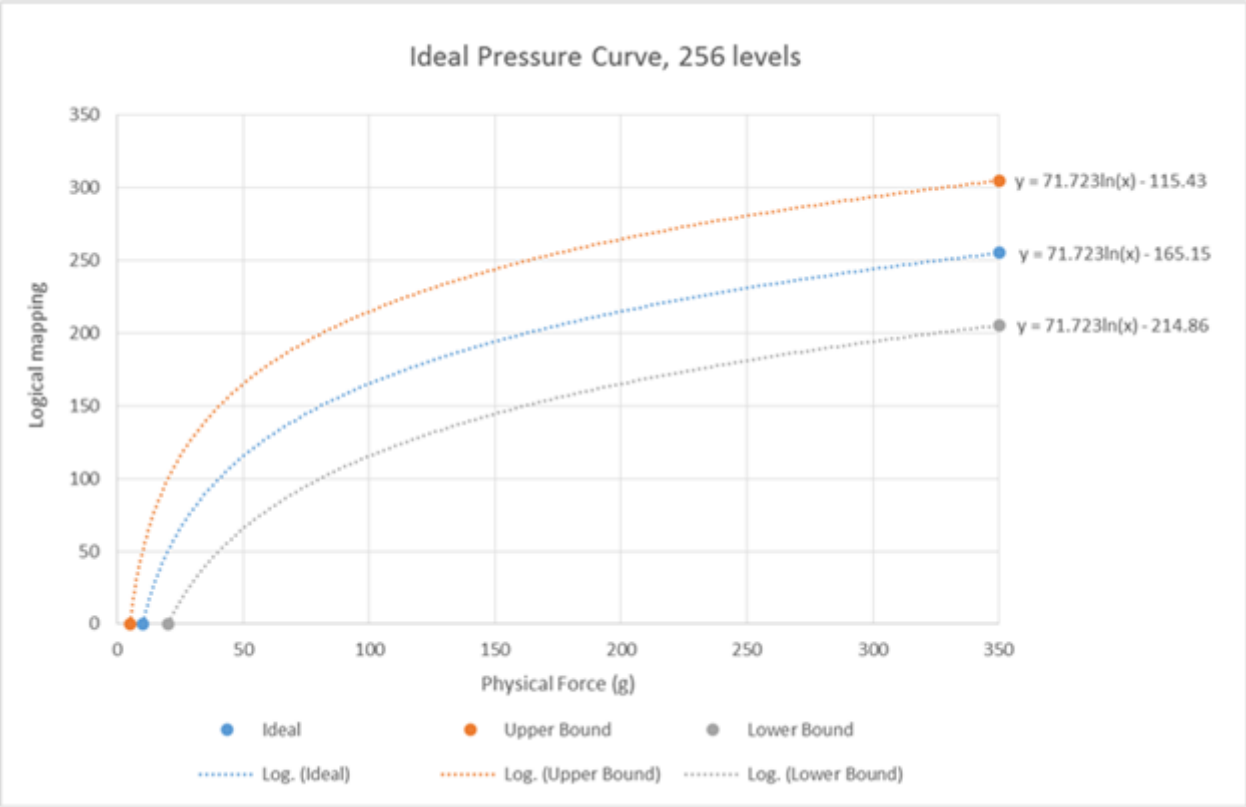
Passing criteria

- 9 out of 11 test iterations must pass for test to succeed.
  - Iterations 1 and 2 are optional and not required to pass.
  - Iterations 3-11 must pass for test to succeed.
- The reported levels of pressure for the corresponding applied weights must follow a logarithmic curve.

Logarithmic regression will calculate an $r^2 > 0.9$, and log regression $r^2 >$ linear regression r^2.

- The reported levels of pressure must fall within the tolerance of the ideal pressure curve. The ideal pressure curve and the upper and lower bounds are shown below, normalized to a 0-1 logical scale.

Here's a diagram showing the ideal pressure curve for a Windows pen device.



Pressure Motion

Article • 01/18/2019

This is to test the consistency of the pressure reporting by a Windows pen device as it moves across a screen.

Test name

- PressureMotion

Core requirements tested

- Device.Input.Digitizer.Pen.Pressure.

Test purpose

- Verifies that the pen has a consistent level of pressure reported across the screen for given input applied force.

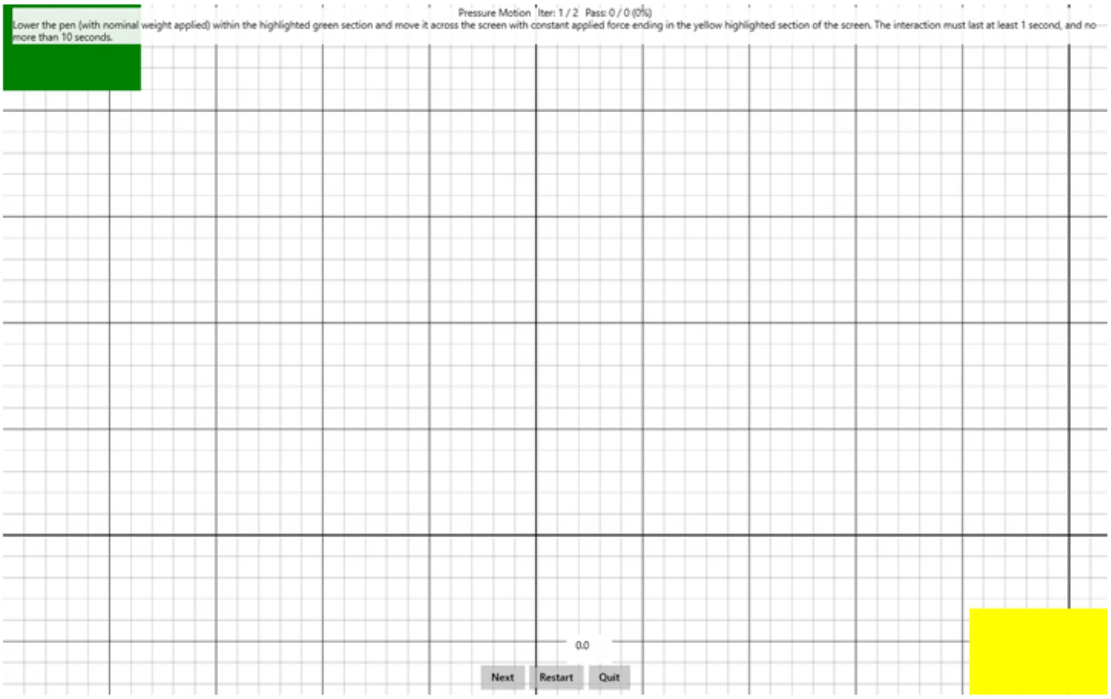
Tools required

- PT3 with Pen holder + 150g weight.

Validation steps

1. Run the test case for Pressure Motion.
2. Following the directions on the screen, lower the pen (which has an applied nominal weight to it) in contact with the screen within the highlighted green section.
    - Ensure that the pen is held in contact with the screen by the weight of the pen, holder, and applied weight, and not by a manual downward force applied to the testing jig.
3. Starting from within the green rectangle, slowly move the pen in contact with the screen, ending within the yellow rectangle. The pen must not be lifted from the screen while in motion, this should last for a minimum of 1 second, and should not take longer than 10 seconds.

Here's a screenshot from the Pressure Motion test.


Pressure Motion Iter: 1 / 2 Pass: 0 / 0 (0%)
Lower the pen (with nominal weight applied) within the highlighted green section and move it across the screen with constant applied force ending in the yellow highlighted section of the screen. The interaction must last at least 1 second, and no more than 10 seconds.
0.0
Next
Restart
Quit

Common error messages

- "Pressure variation is too high."

  Occurs if the reported pressure level is beyond the allowed variance across the screen.

Passing criteria

- 2/2 test iterations must pass for test to succeed.
- The test allows for reported pressure to vary by +/- 5% while moving across the screen.

Resolution

Article • 01/18/2019

This is to test the reporting resolution of a Windows pen device.

Test name

- Resolution

Core requirements tested

- Device.Input.Digitizer.Pen.Resolution.

Test purpose

- Verifies that the pen is able to genuinely report at a resolution that is higher than the display resolution, and also verifies that the reporting resolution is >= 150 DPI.

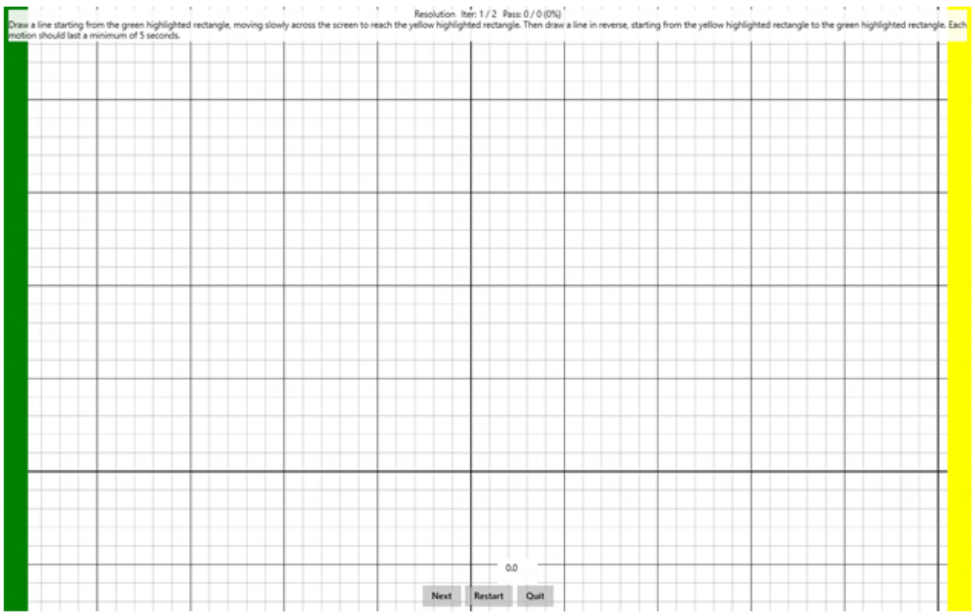
Tools required

- None (PT3 with Pen Holder optional).

Validation steps

1. Run the test case for Resolution.
2. Drag the pen as slowly as possible across the screen from any point in the green highlighted region to any point in the yellow highlighted region without lifting the pen. Then draw a line in reverse, from the yellow highlighted region back to the green highlighted region.
3. All off-axis movement or jitter is ignored, so if the PT3 cannot move smoothly at a slow pace, the pen can be moved by hand across the screen to avoid extra friction.
    - If you use your hand to move the pen across the screen, make sure that your palm does not contact the UI buttons at the bottom of the screen.
4. The pen should always be in contact with the screen during the motion.

Here's a screenshot from the Resolution test.


Resolution Iter: 1 / 2 Pass: 0 / 0 (0%)
Draw a line starting from the green highlighted rectangle, moving slowly across the screen to reach the yellow highlighted rectangle. Then draw a line in reverse, starting from the yellow highlighted rectangle to the green highlighted rectangle. Each motion should last a minimum of 5 seconds.
0,0
Next
Restart
Quit

Common error messages

- "Failed to retrieve screen resolution."

  Occurs if the test fails to retrieve the resolution of the primary display.

- "Resolution too low"

  Occurs if the digitizer's resolution is less than 150 DPI.

- "Pen resolution must be >= screen resolution."

  Occurs if the digitizer's resolution is less than the display resolution.

- "Positional delta too large."

  Occurs if the pen stroke contains too large of a jump between subsequent packets in the axis being tested.

- "Not enough logical coordinates found."

  Occurs if the pen stroke does not cover enough of the digitizer's resolution.

- "The contact did not begin/end in the correct location."

  Occurs if the pen stroke does not begin in the green rectangle and end in the yellow rectangle.

- "The interaction’s duration was too short."

  Occurs if the pen stroke does not last the required duration.

Passing criteria

- 2/2 test iterations must pass for test success.
- Between the 2 slow horizontal swipes, the majority of logical units of X should be hit and reported, and between 2 slow vertical swipes, the majority of logical units of Y should be hit and reported.
- The percentage of logical X or Y coordinates required to be covered in each iteration scales based on the ratio of digitizer resolution to display resolution:
  - If digitizer resolution >= display resolution, hitting 90% of coordinate space is required.
  - If digitizer resolution >= 2x display resolution, hitting 80% of coordinate space is required.
  - If digitizer resolution >= 3x display resolution, hitting 70% of coordinate space is required.
  - If digitizer resolution >= 4x display resolution, hitting 60% of coordinate space is required.
  - If digitizer resolution >= 5x display resolution, hitting 50% of coordinate space is required.

Stationary Jitter (stationary-jitter)

Article • 03/18/2021

This is to test the position reporting accuracy (jitter) of a Windows pen device that is stationary on the screen.

Test name

- StationaryJitter

Core requirements tested

- Device.Input.Digitizer.Pen.Jitter – Stationary Jitter.

Test purpose

- Verifies that while the pen is held perfectly stationary in contact with the screen, the contact position that the device reports is within allowed limits.

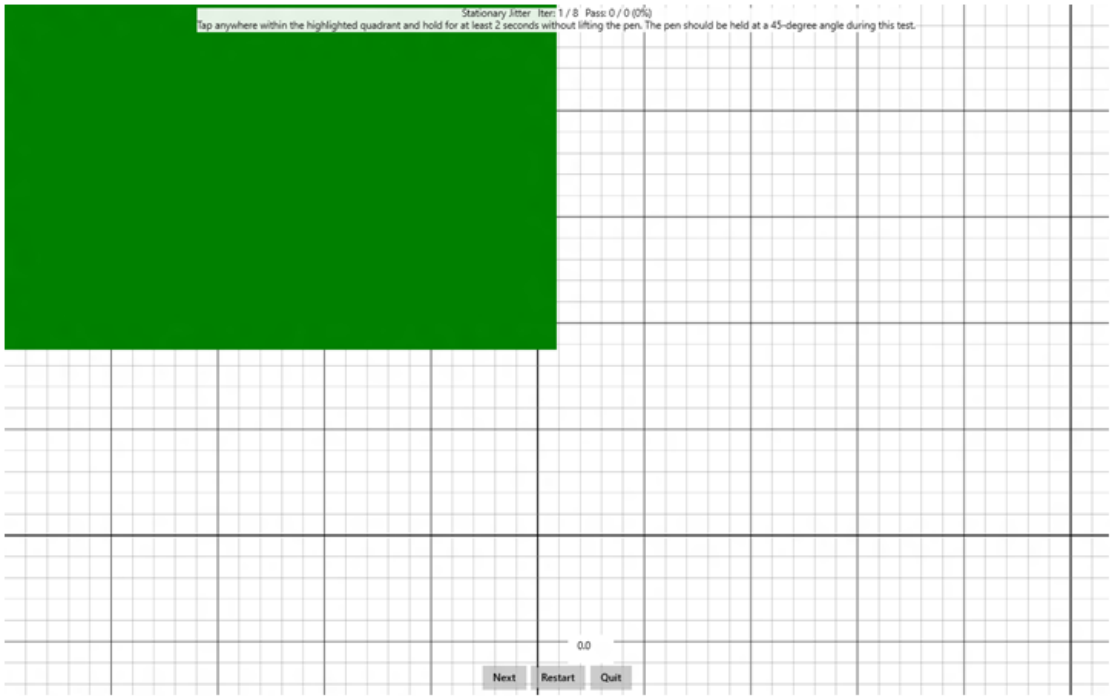
Tools required

- PT3 with Pen holder + 150g.

Validation steps

1. Run the test case for Stationary Jitter.
2. The pen should be held at a 45 degrees angle during this test.
3. Following the directions on the screen, bring pen in contact with screen within the highlighted quadrant.
4. Hold the pen steady in this state for at least 2 seconds.
5. Ensure that activation force at all times when the pen is in contact with the screen.

Here's a screenshot from the Stationary Jitter test.


Stationary Jitter Iter: 1 / 8 Pass: 0 / 0 (0%)
Tap anywhere within the highlighted quadrant and hold for at least 2 seconds without lifting the pen. The pen should be held at a 45-degree angle during this test.
0.0
Next
Restart
Quit

Common error messages

- "Jitter too high."

  Occurs if the pen's jitter exceeds the tolerance.

- "The contact did not begin/end in the correct location."

  Occurs if the pen stroke does not begin and end in the green rectangle.

- "The interaction's duration was too short."

  Occurs if the pen stroke does not last the required duration.

Passing criteria

- 7/8 test iterations must pass for test success.
- This test has a .06 mm (20%) allowed tolerance, therefore a recorded stationary jitter <= 0.36 mm will result in a pass.
- A single iteration that is more than .06 above the allowed tolerance, resulting in a stationary jitter of >=0.42 mm, will fail the entire test.

  Output is derived as the max value of jitter over valid iterations.

Third Party Driver

Article • 05/06/2022

This is to ensure that there is no 3rd party driver for a Windows pen device that is connected via a Windows-supported bus.

Test name

- 3rd Party Driver

Core requirements tested

- Device.Input.Digitizer.Pen.ThirdPartyDrivers.

Test purpose

- Verifies that the pen does not have a 3rd party driver if the device is on a USB, I^2C, or Bluetooth bus.

Tools required

- None.

Validation steps

1. Run the test case for 3rd party drivers.
2. In the dialog box that appears, answer "Yes" if your device is on a USB, I^2C, or Bluetooth bus, otherwise, answer "No."

Note: If wake-on-touch is enabled, answer "No" to the "Is the following input device on an in-box supported bus?" question, even if the input device is on an in-box supported bus. Answering "No" under any other cases when on an in-box supported bus is a violation of policy.

Common error messages

- "Must not require additional drivers."

  Occurs if the pen fails the third party driver requirement.

Passing criteria

- 1/1 test iterations must pass for the test to succeed.

Haptic Pen Validation Guide

Article • 11/05/2021

This article will guide you through using the Haptic Pen Validation Tool to validate both required (**Test Cases 1 and 2**) and optional (**Test Cases 3 and 4**) haptic pen features.

Test Prerequesites

Install the Haptic Pen Validation Tool

1. Download the Haptic Pen Validation Tool
2. Right-click on the downloaded **HapticPenValidationTool.appx** file
3. Select **Properties**
4. Select the **Digital Signatures** tab
5. Select the signature in the list
6. Press the **Details** button
7. Press the **View Certificate** button
8. Press the **Install Certificate...** button
9. Select **Local Machine** and press the **Next** button
10. Select **Place all certificates in the following store** and press the **Browse...** button
11. Select **Trusted People** and press the **OK** button
12. Press the **Next** button
13. Press the **Finish** button
14. Double-click on **HapticPenValidationTool.appx** and press the **Install** button

Test Cases

Follow the steps below to run tast cases for each scenario.

Test Case 1: Verifying the pen properly reports required haptic capabilities

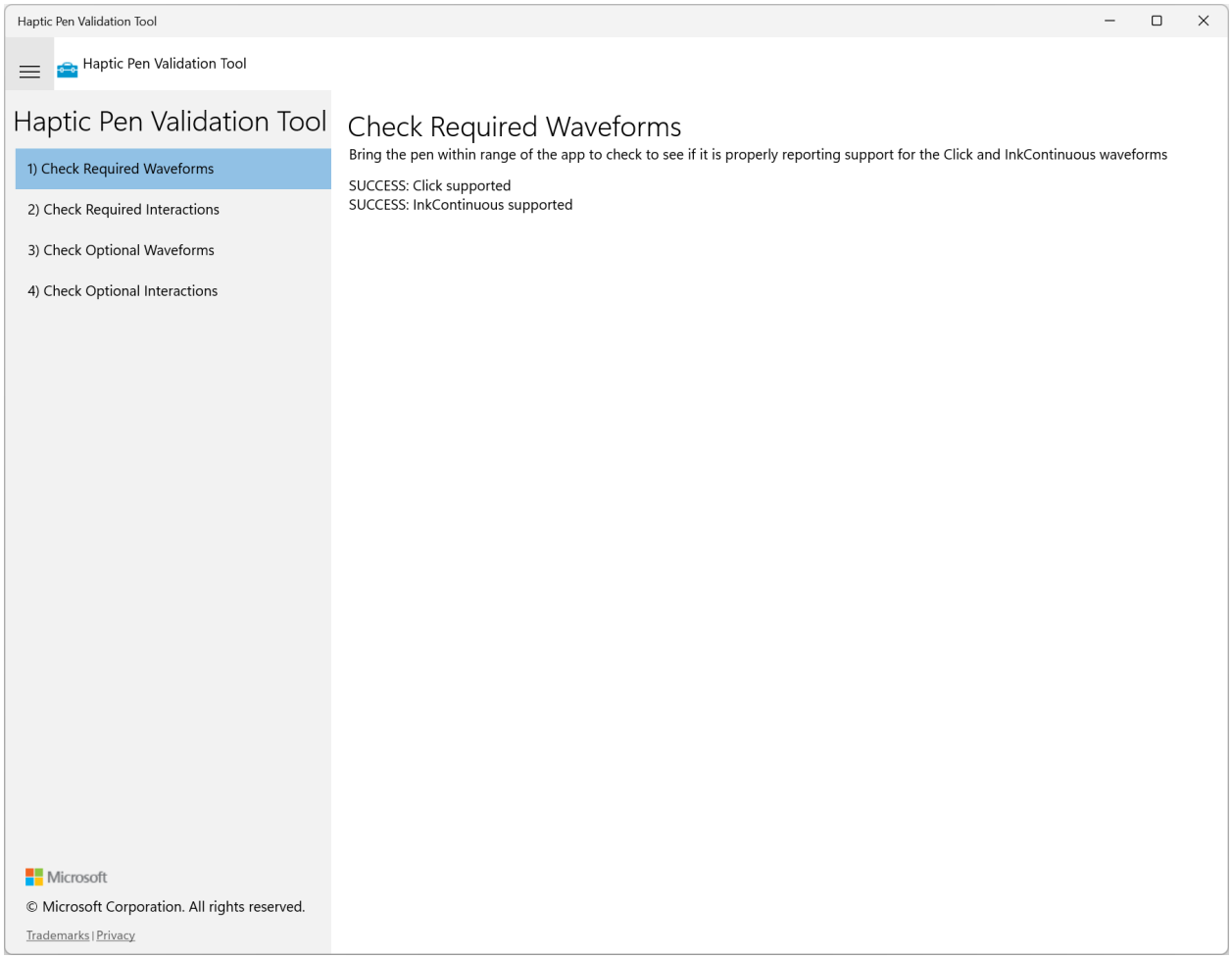
1. Launch the Haptic Pen Validation tool and navigate to the **"Check Required Waveforms"** tab
2. Bring the pen within range of the main canvas

Expected results for Test Case 1

The below text is shown:

SUCCESS: Click supported

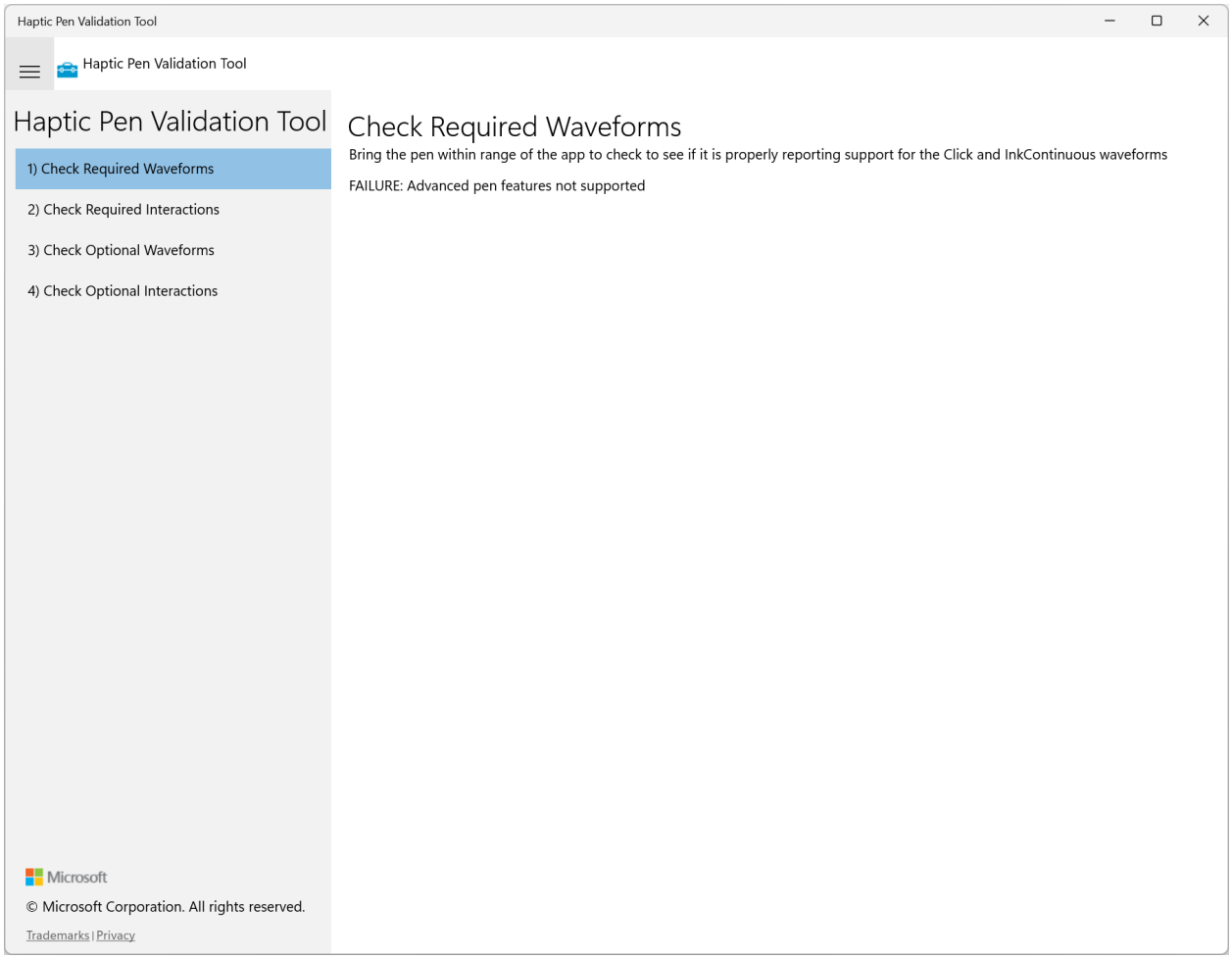
SUCCESS: InkContinuous supported


Haptic Pen Validation Tool
Haptic Pen Validation Tool
Haptic Pen Validation Tool
1) Check Required Waveforms
2) Check Required Interactions
3) Check Optional Waveforms
4) Check Optional Interactions
Check Required Waveforms
Bring the pen within range of the app to check to see if it is properly reporting support for the Click and InkContinuous waveforms
SUCCESS: Click supported
SUCCESS: InkContinuous supported
Microsoft
© Microsoft Corporation. All rights reserved.
Trademarks | Privacy

Failure cases for Test Case 1

Error Case	Error Reason
"FAILURE: Current device is not a pen" text is shown	The current pointer device is not recognized as a pen
"FAILURE: Advanced pen features not supported" text is shown	The current pen does not support advanced features, such as tactile feedback
"FAILURE: This pen does not provide tactile feedback" text is shown	The current pen does not support haptics
"FAILURE: Click not supported" text is shown	The current pen does not support the WAVEFORM_CLICK
"FAILURE: InkContinuous not supported" text is shown	The current pen does not support the WAVEFORM_INKCONTINUOUS

Example failure message:


Haptic Pen Validation Tool
Haptic Pen Validation Tool
Haptic Pen Validation Tool
1) Check Required Waveforms
2) Check Required Interactions
3) Check Optional Waveforms
4) Check Optional Interactions
Check Required Waveforms
Bring the pen within range of the app to check to see if it is properly reporting support for the Click and InkContinuous waveforms
FAILURE: Advanced pen features not supported
Microsoft
© Microsoft Corporation. All rights reserved.
Trademarks | Privacy

Test Case 2: Verifying required interactions

1. Successfully pass **Test 1**
2. Navigate to the **"Check Required Interactions"** tab of the validation tool
3. Use the pen to tap the **"Send Click Feedback"** button
4. Place the pen in the box in the **InkContinuous Test** and move it for more than one second while keeping the tip in contact with the screen
5. Place the pen in the box in the **Inking and Interaction Feedback Test** and move it for one second while keeping the tip in contact with the screen
6. Once the Click waveform is felt, hold the pen in place with the pen continuing to touch the screen but not moving for one second
7. Continue moving the pen in the box

Expected results for Test Case 2

1. The "Click" waveform is played by the pen upon tapping the button in Step 3.
2. The "InkContinuous" waveform is played, starting when the pen tip comes down in Step 4 and then stops after one second of the pen being in contact with the screen.

3. The “InkContinuous” waveform is played, starting when the pen tip makes contact with the screen in Step 5.
4. The “Click” waveform is played after one second of dragging the pen in the box in Step 5.
5. The “InkContinuous” waveform is played once the pen is moved in Step 7.

Failure cases for Test Case 2

Error Case	**Error Reason**
No haptic feedback is felt when the button is tapped with the pen in **Step 3**	The pen did not properly play the WAVEFORM_CLICK waveform
No haptic feedback is felt when the pen is moved during **Step 4**	The pen did not properly play the WAVEFORM_INKCONTINUOUS waveform
Haptic feedback does not stop after one second of movement during **Step 4**	The pen did not properly respond to WAVEFORM_STOP
No haptic feedback is felt when the pen is moved during **Step 5**	The pen did not properly play the WAVEFORM_INKCONTINUOUS waveform
The “Click” waveform is not played after one second of movement in **Step 5**	The pen did not properly play the WAVEFORM_CLICK waveform
No haptic feedback is felt when the pen is moved during **Step 7**	If haptic feedback was felt during **Step 3** and **Step 5**, this indicates that the pen does not properly resume an Inking waveform when interrupted by an Interaction waveform. If not, this indicates that that the pen did not properly play the WAVEFORM_INKCONTINUOUS waveform.

Test Case 3: Verifying optional waveforms

The tests included in this section are optional and should only be used if the current pen supports the relevant waveforms.

1. Successfully pass **Test 1** and **Test 2**
2. Navigate to the "**Check Optional Waveforms**" tab of the validation tool
3. Choose a Continuous waveform to test in the dropdown in the **Optional Continuous Waveforms** section
4. Place the pen in the box in the **Optional Continuous Waveforms** section and move around
5. Lift the pen from the screen
6. Choose an Interaction waveform to test in the dropdown in the **Optional Interaction Waveforms** section
7. Use the pen to tap the "**Send Interaction Feedback**" button

Expected results for Test Case 3

1. The chosen waveform is played, starting when the pen tip comes down in **Step 4** and then stops when the pen is lifted in **Step 5**.
2. The chosen waveform is played by the pen upon tapping the button in **Step 7**.

Failure cases for Test Case 3

Error Case	Error Reason
No haptic feedback is felt when the pen is moved during **Step 4**	The pen did not properly play the chosen waveform
Haptic feedback does not stop when lifted from the screen in **Step 5**	The pen did not properly react to the pen tip being lifted from the screen by stopping the waveform playback
No haptic feedback is felt when the button is tapped with the pen in **Step 7**	The pen did not properly play the chosen waveform

Test Case 4: Verifying optional interactions

The tests included in this section are optional and should only be used if the current pen supports the relevant features.

1. Successfully past **Test 1** and **Test 2**
2. Navigate to the "**Check Optional Interactions**" tab of the validation tool

3. Use the pen to press the **"Send Click Feedback With Intensity"** button
4. Change the **Intensity** slider
5. Use the pen to press the **"Send Click Feedback With Intensity"** button again
6. Place the pen in the left box in the **Automatically Stop Continuous Feedback Test** and move it around
7. Lift the pen out of range from the screen
8. Place the pen tip down in the right box in the **Automatically Stop Continuous Feedback Test** and move it around
9. Use the pen to press the **"Send Repeated Feedback"** button
10. Use the pen to press the **"Send Repeated Feedback With One Second Pause"** button

Expected results for Test Case 4

1. The Click waveform is played at 100% intensity when the button is pressed in **Step 3**.
2. The Click waveform is played at the chosen intensity when the button is pressed in **Step 5**.
3. The "InkContinuous" waveform is played, starting when the pen tip makes contact with the screen in **Step 6**.
4. The "InkContinuous" waveform stops when the pen is lifted from the screen in **Step 7**.
5. The "InkContinuous" waveform does not play when the pen tip placed in the right box and moved around in **Step 8**.
6. The Click waveform is played three times when the button is pressed in **Step 9**.
7. The Click waveform is played three times with a one second pause between each when the button is pressed in **Step 10**.

Failure cases for Test Case 4

Error Case	Error Reason
The Click waveform did not play at the chosen intensity when the button is pressed in **Step 5**	The pen did not properly implement the Intensity feature
No haptic feedback is felt when the pen is moved during **Step 6**	The pen did not properly play the WAVEFORM_INKCONTINUOUS waveform
Haptic feedback does not stop when lifted from the screen in **Step 7**	The pen did not properly react to the pen tip being lifted from the screen by stopping the waveform playback

Error Case	Error Reason
The InkContinuous waveform is played when the pen is placed during **Step 8**	The pen did not properly clear its waveform state when leaving range of the screen
The Click waveform did not play for three times when the button is pressed in **Step 9**	The pen did not properly implement the Repeat Count feature
The Click waveform did not have a one second pause between each playback when the button is pressed in **Step 10**	The pen did not properly implement the Retrigger Period feature

Appendix (appendix)

Article • 03/18/2021

The appendix for the Windows Pen Validation Guide, provides information about manually running the Windows Hardware Lab Kit (HLK) tests, some points to note (following our latest content release), and answers to some frequently asked questions.

Manual run

Depending on your testing situation, it may be necessary to run tests manually. This is also referred to as running the tests “standalone” from the command line, as opposed to running the tests through the HLK Studio. Tests that are run manually cannot be submitted for compatibility certification. So you should only run tests manually, if you have an issue that prevents you from running them through HLK Studio, or you have been instructed to do so by the Pen HLK team.

If you need to contact the Pen HLK team, you can reach them via email at pen_qn@microsoft.com.

Before Running Input HLK Tests

Before running the tests standalone, please make sure that you have installed the normal HLK client on the device under test, and you've obtained a developer license that is still current. Installing the HLK client will ensure that all needed prerequisites for testing are installed on the device. After the HLK client is installed and the developer license is acquired, you should be able to run all the input HLK tests standalone.

Files required to run test

The pen test files can be found in the following locations:

- On an x86 system:

  C:\Program Files (x86)\Windows Kits\10\Hardware Lab Kit\Tests**x86**\input\Digitizer

- On an amd64 system:

  C:\Program Files (x86)\Windows Kits\10\Hardware Lab Kit\Tests**amd64**\input\Digitizer

Make sure that the pen test folder contains the files required for the TAEF test framework, as well as files for pen-specific tests. Here's a list of files that you would typically find in the pen tests folder:

- Core pen tests
  - PenTests.dll
  - PTS.dll
  - Pen.xml
  - PenLatencyTests.wsc
  - PenStaticTests.wsc
- Pen test tools
  - audiotouch.exe
  - stepmotor.exe
- Test harness files
  - TAEF folder
- Supporting files
  - wlklogannotation.dll
  - WTTlog.dll

Running all Tests in PenTests

Use the following command to run all the tests contained in PenTests:

```
taef\\te.exe PenTests.dll
```

Running specific tests in PenTests

To run a specific test, use the **/name** parameter, using the table ID given in pen.xml:

```
taef\\te.exe PenTests.dll /name:\*ContactAccuracy
```

Running Latency tests using RA Tool

- Running all Tests that use the RA Tool

  Use the following command to run all tests contained in RA tool tests (audiotouch/stepmotor):

```
taef\\te.exe PenLatencyTests.wsc
```

- Running specific tests that use RA Tool

  To run a specific test, use the **/name** parameter. For example:

```
taef\\te.exe PenLatencyTests.wsc /name:\*audiotouch
taef\\te.exe PenLatencyTests.wsc /name:\*stepmotor
```

Running test for third party drivers (static test)

The following command line will run static test for third party drivers:

```
taef\\te.exe PenStaticTests.wsc
```

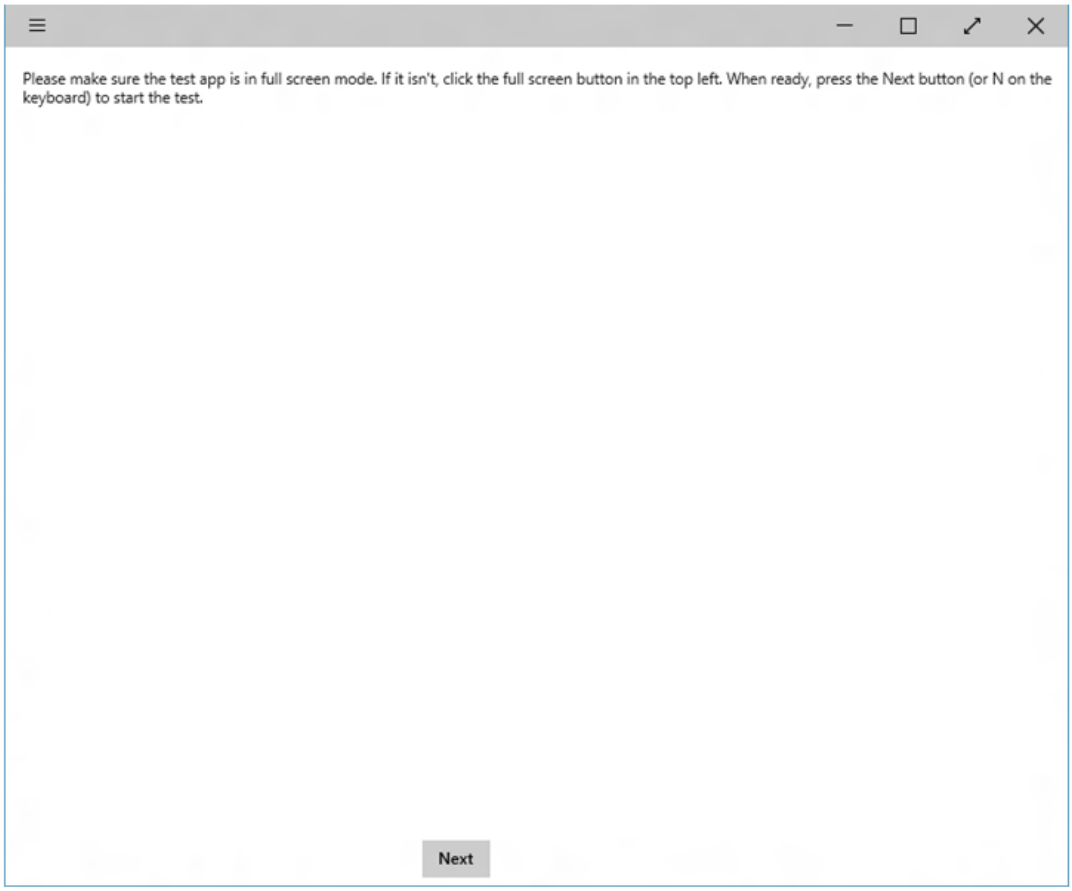
Here's a screenshot of the result of running the static test:

StartGroup: PenStaticTests::ThirdPartyDrivers
Check for InboxDrivers vs third party driver
DUT : HID-compliant pen
Verify: AreEqual: Must not require additional drivers - Values (0, 0)
EndGroup: PenStaticTests::ThirdPartyDrivers [Passed]
Summary: Total=1, Passed=1, Failed=0, Blocked=0, Not Run=0, Skipped=0

Release notes

Initial step to start running PenTests

The first time *PenTests.dll* is run, it may require a switch to full screen mode. Please follow the steps on the screen, to switch to full-screen mode. Here's a screenshot of the instructions for switching:


Please make sure the test app is in full screen mode. If it isn't, click the full screen button in the top left. When ready, press the Next button (or N on the keyboard) to start the test.
Next

For some devices, full screen mode may not function properly

For some devices, the full screen window may not cover the entire screen, and the test background canvas may appear to be stretched. This can be resolved by setting the scaling to 100%.

Frequently asked questions

- How can partners submit questions about the pen HLK and testing process?

  Any questions about the Pen HLK testing and validation process can be directed to the Ask Windows 10 Pen alias (pen_qn@microsoft.com).

- For a pen device that works on different platforms, do I need to repeat the pen HLK test, and resubmit for each system that uses the pen?

  Yes, our HLK certification process is done per integrated system. We do not certify standalone pen devices.

- Can a passive stylus or any other types of pens get tested by HLK?

Only active pen devices that conform to the Active Pen Implementation Guide will be capable of completing HLK testing.

- I have a device that is running Windows 10 Mobile. Can I still use the HLK for compatibility testing?

  HLK tests for devices running Windows 10 Mobile are not available at this time. Until the tests are available, we will be relying on our hardware partners to attest to the fact that their devices meet the Windows 10 requirements.

- I have a device with a diagonal screen size that is smaller than 4.5" or larger than 30". Can I still use the HLK for compatibility testing?

  Unfortunately, the assistive testing jigs and some of the HLK tests only support devices with screen sizes from 4.5"-30". If your device does not meet these size requirements, please contact us directly and we will work with you to set up a testing process so you can receive compatibility certification.

Precision Touchpad Devices (touchpad-devices)

Article • 01/24/2024

This section provides information about how to implement and validate a Windows Precision Touchpad device, for Windows 10 and later operating systems.

In this section

Expand table

Topic	Description
Precision Touchpad Implementation Guide	This section is an implementation guide for a Windows Precision Touchpad device, for Windows 10 and later operating systems.
Precision Touchpad Validation Guide	This section is a validation guide for a Windows Precision Touchpad device, for Windows 10 and later operating systems.

Precision Touchpad Implementation Guide (touchpad-implementation-guide)

Article • 01/24/2024

This section is an implementation guide for a Windows Precision Touchpad device, for Windows 10 and later operating systems.

The information includes guidance on using the Human Interface Device (HID) protocol to communicate with a Windows host.

In this section

Expand table

Topic	Description
Device Bus Connectivity	This topic discusses bus connectivity methods for a Windows Precision Touchpad device.
Power Management	This topic describes power management for a Windows Precision Touchpad implementation.
Protocol Implementation	Windows Precision Touchpads are expected to use the Human Interface Device (HID) protocol to communicate with the host. This topic describes how to implement the HID protocol for Windows Precision Touchpads.
Device Integration	Windows Precision Touchpads define an experience. Integration of the module has a significant impact on how well that experience is realized. This topic describes Windows Precision Touchpads integration.
Module Design for Windows HCK Requirements	The Windows Hardware Certification Kit (HCK) requirements for Windows Precision Touchpads are designed to provide a consistent user experience where precision and reliability are at the forefront. These requirements shall influence all aspects of the module, including the sensor, controller IC and associated mechanics.

Topic	Description
Enable or Disable Toggle Button	Windows Precision Touchpads (or legacy touchpads that have opted in for enable/disable control in Windows 8.1) can have their enable/disable state toggled by using a hardware button or keyboard combination.
Windows Precision Touchpad Industrial Design Guide	This topic provides industrial design guidance for Windows Precision Touchpads. Many of these guidelines are part of the Windows Hardware Certification Kit (HCK) and are identified as such in this topic. Additional industrial design characteristics are provided as suggested guidance and reference for hardware manufacturers. This information applies to Windows 8.1.
Legacy Touchpad PC Settings Opt-In	This topic describes how devices that are not Windows Precision Touchpads can opt in to various settings that are exposed in Windows 8.1 to provide a simple and easy-to-navigate inbox solution to manage the most common touchpad settings.

Glossary

Expand table

Term/Acronym	Definition
Global Items	Entries in a report descriptor that describe the properties of a data field. Unlike local items, global items remain the same across multiple main items until they are changed.
Human Interface Device (HID)	Human interface device (HID) refers to either the protocol or the device itself. In this document, HID refers to the device and HID protocol refers to the protocol in definition.
I²C - I²C (Inter-Integrated Circuit)	I²C is a multi-master serial single-ended bus that is used to attach low-speed peripherals to the motherboard, embedded.
Local Items	Entries in a report descriptor that describe certain properties of a data field. Local items are only associated with the main items they precede in the report descriptor.
Main Items	Entries in a report descriptor that define data fields.

Term/Acronym	Definition
Usage	The name of a value, button, or collection in a HID report.
USB	Universal Serial Bus

Related topics

Windows pointer device data delivery protocol

USB HID Information

HID I2C v1.0 Protocol Specification

Advanced Configuration and Power Interface Specification

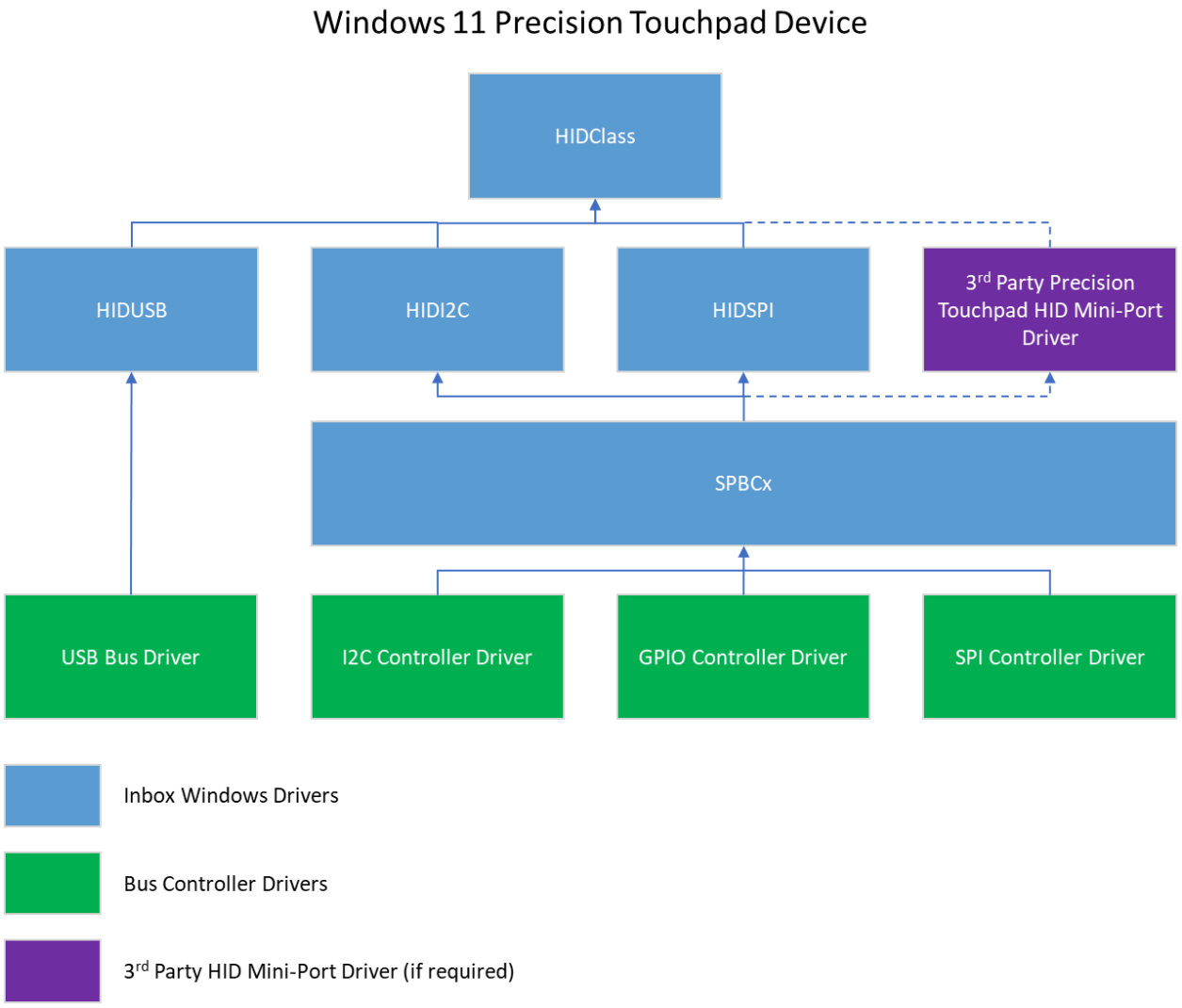
Device Bus Connectivity (touchpad-device-bus-connectivity)

Article • 06/21/2022

> **Note**
>
> This article contains references to the term *slave*, a term that Microsoft no longer uses. When the term is removed from the software, we'll remove it from this article.

This topic discusses bus connectivity methods for a Windows Precision Touchpad device.

A Windows Precision Touchpad device can use the Microsoft-provided inbox drivers to connect to its Windows host, using either HID over USB, HID over I^2C, or (starting in Windows 11) HID over SPI. However you can use any other bus that you want, as long as you provide the required Windows-compatible, 3rd party HID mini-port driver for the touchpad device.



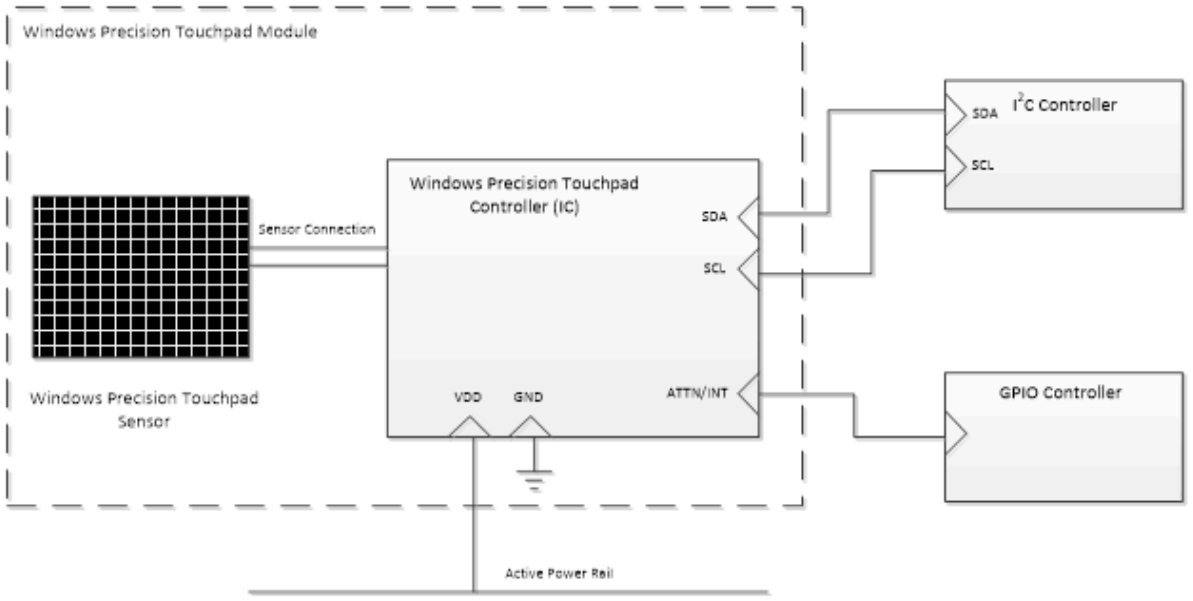
I²C Devices

A Windows Precision Touchpad module is defined as the combination of a controller IC, a sensor and any associated mechanisms.

A Windows Precision Touchpad module that is connected to its Windows host via the I²C bus must, at a minimum, expose the following five connection pins:

- A data line (SDA)
- A clock line (SCL)
- An interrupt line
- A power supply line
- A ground connection (GND)

The following diagram shows the I²C connecting signal lines between a Windows Precision Touchpad and its Windows host.



When connecting to an I²C controller it is important to understand the bandwidth demands of all components sharing that controller. A minimum I²C clock speed of 400 KHz is recommended for a Windows Precision Touchpad module. It is highly recommended that touch screen controllers and Windows Precision Touchpad controllers do not share the same I²C controller, as this may result in bandwidth demands that exceed bus capability.

It is recommended that the interrupt line (also referred to as the ATTN line) is connected to an On-SoC GPIO controller or an IOAPIC. If wake functionality is desired, the GPIO or IOAPIC resource to which the interrupt line is connected, must be capable of waking the

SoC in order to allow the Windows Precision Touchpad to wake the system in various scenarios.

If you decide to use the wake functionality, the power line that is connected to the Windows Precision Touchpad module should not be shared with other devices which are not wake-capable. In order for wake scenarios to function successfully, the power line that is used must be energized during connected standby or S3 conditions.

ACPI Table Entries

A Windows Precision Touchpad device that is connected to its Windows host via the I^2C bus must define an entry in the ACPI table for the device to be recognized. The entry should specify the following information:

Expand table

Entry	Description
ACPI Device Entry Name	A 4-character identification unique to this ACPI table, to reference the device. For example, "TPAD".
ACPI Hardware ID	A 4-character + 4-number ACPI hardware ID, to reference the device. The ACPI Hardware ID will exposed in device manager. For example, "MSFT0001".
Compatible ID	This should always be “PNP0C50” to indicate that the device is HID I^2C-compatible.
I^2C Controller	The I^2C controller to which the device is connected, must be specified in order to communicate with it. For example, "I2C3" to represent controller #3).
I^2C Slave Address	The I^2C slave address for the device must be specified to allow the host to address it. For example, "0x6F".
I^2C Speed	The maximum speed supported by both the device and the I^2C controller should be specified here to ensure good communications performance. This should not be any lower than 400KHz (or 0x61A80 in hexadecimal).
GPIO Controller	The GPIO controller to which the interrupt line of the device is connected, must be specified to allow the host to access it. For example, "GPIO0" – for controller zero (0).
GPIO Resource or Pin	The actual pin on the GPIO controller to which the interrupt line of the device is connected, must be specified to allow the host to associate the pin with the device. For example, "{35}" to represent pin 35).
GPIO Resource Type	This defines the constraints around the GPIO resource. This should be "Exclusive" unless SoC wake is desired, in which case, this should be "ExclusiveAndWake".

Entry	Description
GPIO Interrupt Assertion Type	This defines whether the device will provide edge triggered, or level triggered interrupts. HID I²C-compliant devices should provide and indicate "Level" triggered interrupts.
GPIO Interrupt Assertion Level	This defines the line level when the interrupt is asserted by the device. This can be specified as "ActiveLow" or "ActiveHigh".

USB Devices

A Windows Precision Touchpad module that is connected to its Windows host via high-speed/full-speed USB 2.0, should expose the necessary pins for host connectivity.

Connection to the host can take many forms and is at the discretion of the integrator.

Note that when connecting to a USB hub it is important to understand the bandwidth demands of all components sharing that hub. It is highly recommended that high bandwidth devices and Windows Precision Touchpad controllers do not share the same USB hub as this may result in bandwidth demands that exceed bus capability.

USB Bridge Devices (I²C -> USB)

If you decide to use a USB bridge for connecting an I²C Windows Precision Touchpad to its Windows host, then the bridge should expose the touchpad as a distinct device node with the device's unique attributes (wVendorID, wProductID, wVersionID).

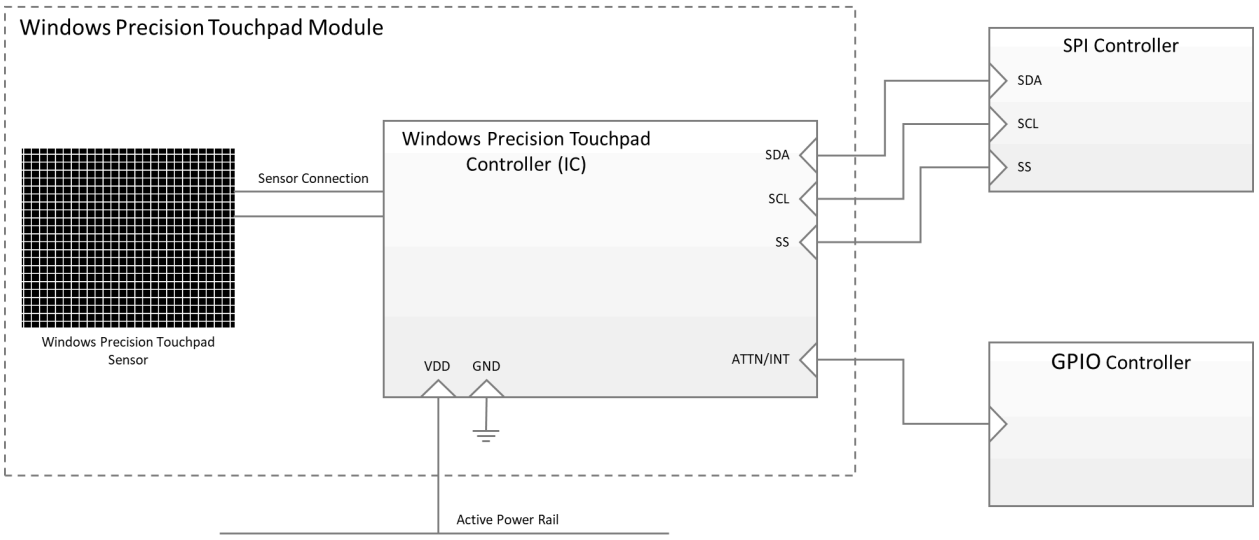
HID SPI Devices

An Windows Precision Touchpad module is defined as the combination of a controller IC and a sensor.

A Windows touchpad module that connects to its Windows host via the SPI bus must, at a minimum, expose the following five connection pins:

- A data line (SDA)
- A clock line (SCL)
- A chip select line (SS)
- An interrupt line
- A power supply line
- A ground connection (GND)

The following is a diagram of the connections lines between a Windows Touchpad device and its Windows host.



ACPI Table Entries

A Windows Precision Touchpad device that is connected via SPI must define an entry in the Advanced Configuration and Power Interface (ACPI) table of the host, for the device to be recognized by the host. For more information about ACPI, see Advanced Configuration and Power Interface Specification.

The ACPI table entry should specify the information as described in the Plug and Play support for HID over SPI article.

Power Management (touchpad-power-management)

Article • 01/24/2024

This topic provides information about power management for a Windows Precision Touchpad device, and also discusses power consumption requirements.

Power consumption

Power consumption requirements for the various power modes of an integrated Windows Precision Touchpad are at the discretion of the OEM or system builder. However, Windows provides facilities to help balance power efficiency and response latency.

0.9 x (IDLE Power Consumption in mA) + 0.1 x (Active Power Consumption in mA) <= 25

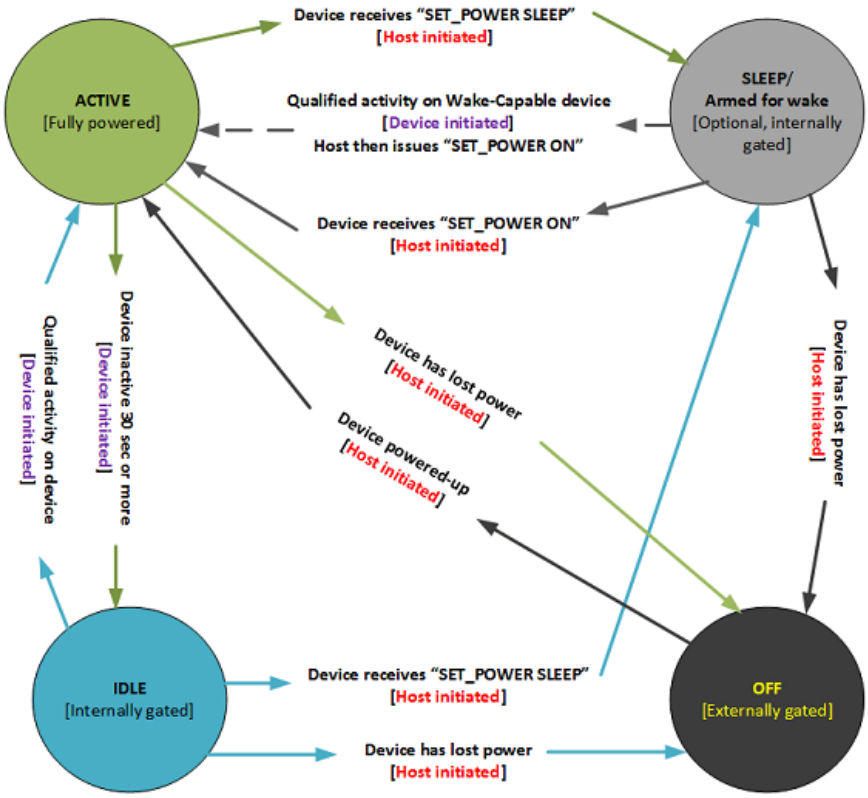
Power consumption for the Sleep mode of a Windows Precision Touchpad (irrespective of bus connectivity) is required to be <= 1mW.

I²C Devices

Windows Precision Touchpad devices that connect to their Windows hosts via I²C, can implement support for up to 4 distinct power states:

- Active
- Idle
- Either Sleep, or Armed for Wake (both optional)
- Off

The four power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode when one or more contacts are present, the button is down, or there has been activity within 30 seconds. When power is applied to a device, then after device booting is completed, the device should be ready and in the **Active** power state.

A device should adhere to the contact down latency and contact move latency requirements for this mode, as this is where the bulk of compatibility requirements will be tested, and where the majority of user interaction will occur.

Idle State

The **Idle** state is defined as the device operating mode when no activity has occurred within 30 seconds.

A device can use reduced scan rate in this mode to reduce power consumption, while still adhering to the contact down latency requirement for this mode.

Once the device has detected activity, it should transition back to the **Active** state.

Sleep (Armed for Wake) State

The **Sleep** state is defined as the device operating mode when the device has been issued a HID I^2C SET_POWER SLEEP command by the host.

In the Sleep state, a device shall consume no greater than 1mW. A device can use reduced scan rate in this mode to significantly reduce power consumption, while still being capable of asserting an interrupt in response to qualified activity, to wake the system. A Windows Precision Touchpad device should ensure that interrupts are not asserted for spurious contacts, since such contacts would result in an unintended system wake. There are no contact down latency requirements for this mode. However, it is recommended that continuous contact that lasts longer than one second should result in an interrupt being asserted.

The device should transition to the **Active** state after receiving a HID I^2C SET_POWER ON command from the host.

Off State

The **Off** state is defined as the device operating mode when the device has had its power completely removed. When power is applied to the device, then after device booting is completed (which shall take no longer than 100ms), the device should be ready and in the **Active** state. Device boot-up should complete before the computer display is initialized and enabled.

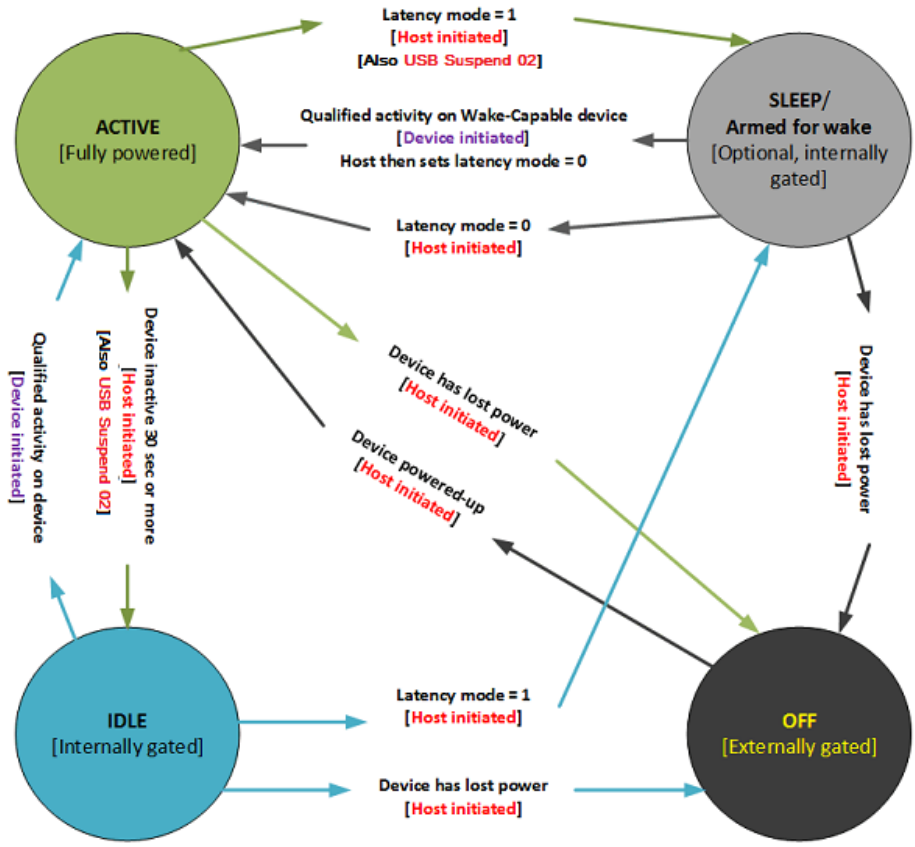
In the **Off** state a device should not consume any power.

USB Devices

Windows Precision Touchpad devices that connect to their Windows hosts via USB, can implement support for up to 4 distinct power states:

- Active
- Idle
- Either Sleep, or Armed for Wake (both optional)
- Off

The four power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode when the host has not suspended the device. When power is applied to a device, then after device booting is completed, the device should be ready in the **Active** power state.

A device shall adhere to the contact down latency and contact move latency requirements for this mode.

Idle State

The **Idle** state is defined as the device operating mode when no contacts or button activity has occurred within a host defined period, and the device has therefore been suspended. This is referred to as *USB selective suspend*.

Windows Precision Touchpad devices that connect via USB should support selective suspend, and report this capability via a Microsoft OS descriptor.

A device can use reduced scan rate in this mode, to reduce power consumption while still adhering to the contact down latency requirement for this mode.

Once the device has detected a qualified activity, it should signal a remote wake. From the time the remote wake event is detected, the device should buffer at least 100ms worth of contact reports, to ensure that little to no input is lost while the USB host controller is resuming.

Sleep (Armed for Wake) State

The **Sleep** state is defined as the device operating mode when the host has transitioned to S3, or Connected Standby. This is indicated to the device via the latency mode feature report, with a value of '1' indicating that maximum latency is permitted. The device should exit this high latency mode both on activity and on host resume.

In the Sleep state, a device shall consume no greater than 1mW. A device can use reduced scan rate in this mode, to significantly reduce power consumption while still being capable of signaling a remote wake as qualified activity to wake the system. A Windows Precision Touchpad should ensure that remote wake is not signaled for spurious contacts, since such contacts would result in an unintended system wake. There are no contact down latency requirements for this mode. However, it is recommended that continuous contact that lasts longer than one second, should result in an interrupt being asserted. A contact occurrence that causes entry into the sleep state should not be reported to wake the system.

Off State

The **Off** state is defined as the device operating mode when the device has had its power completely removed. When power is applied to the device, then after device booting is completed (which shall take no longer than 250ms), the device should be ready and in the **Active** state. Device boot-up should complete before the computer display is initialized and enabled.

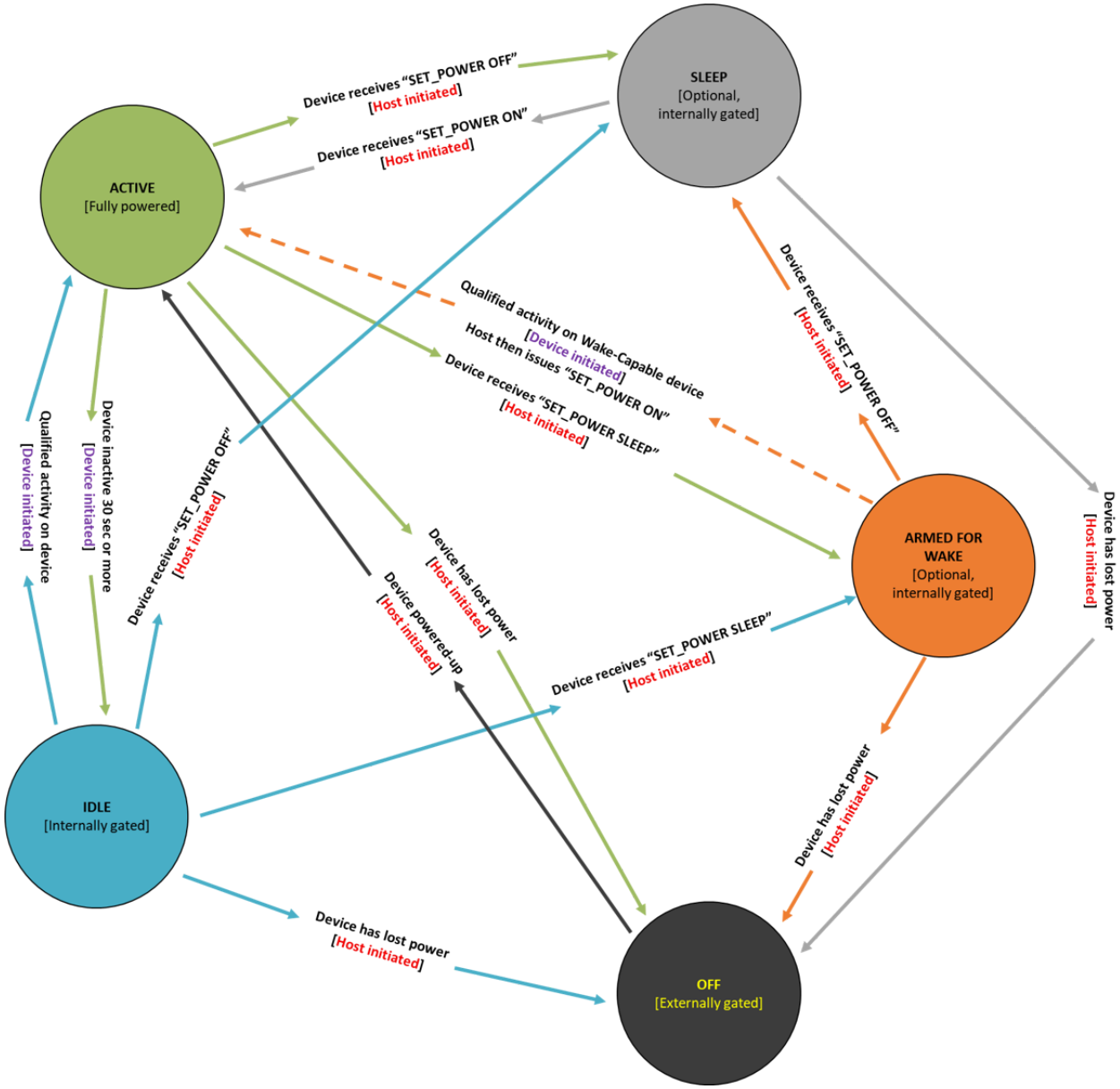
In the **Off** state a device should not consume any power.

SPI Devices

Windows Precision Touchpad devices that are connected via SPI can implement support for up to five distinct power states:

- Active
- Idle
- Sleep (optional)
- Armed for Wake (optional)
- Off

The five power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which one or more contacts are present, or there has been activity within the last 30 seconds. When power is applied to a Precision Touchpad device, then after device boot-up is completed, the device should be ready and in the **Active** power state.

A Precision Touchpad device should adhere to the *contact down latency* and *contact move latency* requirements for this mode, as this is where the bulk of the compatibility requirements will be tested, and where the majority of user interactions will occur.

Idle State

The **Idle** state is defined as the device operating mode in which no activity has occurred within the last 30 seconds.

A device can elect to reduce its scan rate in this mode, to reduce power consumption while still adhering to the contact down latency requirement for this mode. Once the device has detected activity, it should transition back to the **Active** state.

Sleep State

The **Sleep** state is defined as the operating mode that the device enters, after the host sends a HID SPI "SET_POWER OFF" command to the device.

In this state, the device should enter its lowest possible internal power state and should not be scanning for input or be armed to wake.

The device should transition to the **Active** state after it receives a HID SPI "SET_POWER ON" command from the host.

Armed for Wake State

The **Armed for Wake** state is defined as the device operating mode in which the host has transitioned to a lower power state, but is still scanning for input, so it is configured to be woken up by touchpad interactions.

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of signaling a remote wake (in response to qualified activity) to wake the system. A Windows Precision Touchpad device should ensure that remote wake is not signaled in response to spurious contacts, which would result in an unintended system wake. There are no contact down latency requirements for this mode. It is recommended that a continuous contact of more than 1 second should result in signaling a remote wake.

A contact occurrence that causes entry into the armed for wake state should not be reported to wake the system.

The device should transition to the **Active** state after it receives a HID SPI "SET_POWER ON" command from the host.

Off State

The **Off** state is defined as the operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed (which shall take no longer than 250ms), the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

In the **Off** state a device should not consume any power.

Protocol Implementation (touchpad-protocol-implementation)

Article • 10/11/2021

This section provides guidelines for implementing the communication protocol that is used by Windows Precision Touchpad devices.

Touchpad devices are expected to use the Human Interface Device (HID) protocol to communicate with their Windows host.

A good understanding of the HID protocol is needed to be able to understand the information presented here. See the following resources for information about the HID protocol:

- Device Class Definition for Human Interface Devices (HID) Version 1.11
- HID Usage Tables Version 1.12
- HID Over I2C Protocol Specification Version 1.0

Windows includes a HID class driver and corresponding HID I^2C, HID USB, HID SPI, and HID BTH miniport drivers. So there is no need to develop 3rd-party drivers for Windows Precision Touchpad devices that use one of these buses. However, if you want to use a 3rd-party bus controller, or develop your own to use with your device, you can do that.

If you want your device to connect via an alternate bus, then a 3rd-party HID mini-port driver might be required. You only need to report the usages described in this topic, in the firmware for your device. Windows will use your firmware and its own HID drivers to enable mouse and gesture capabilities for your device, and also provide Windows applications with access to your device.

A sample descriptor is provided in the Sample Report Descriptors topic.

In this section

Expand table

Topic	Description
Required HID Descriptors	This topic presents the required HID descriptors (and device attributes) for a Windows Precision Touchpad device in Windows 10 and later operating systems.

Topic	Description
Required HID Top-Level Collections	This section discusses the required HID top-level collections that are used for precision touchpad reporting in Windows 10 and later operating systems.
Sample Report Descriptors	This topic presents a sample report descriptor and sample descriptor excerpts for a Windows Precision Touchpad device.

Required HID Descriptors (touchpad-required-hid-descriptors)

Article • 03/18/2021

This topic presents the required HID descriptors (and device attributes) for a Windows Precision Touchpad device in Windows 10 and later operating systems.

Required USB HID Descriptor

The following table shows the required USB HID descriptor. For more information, see **section 6.2.1** in Device Class Definition for Human Interface Devices (HID) Version 1.11.

Expand table

Member	Size in bytes	Description
bLength	1	Size of the descriptor
bDescriptorType	1	Type of descriptor
bcdHID	2	HID version number
bCountryCode	1	Country code
bNumDescriptors	1	Number of descriptors
bDescriptorType	1	Descriptor type
bDescriptorLength	2	Length of the descriptor

Required I²C HID Descriptor

The following table shows the required I²C HID descriptor.

Expand table

Member	Size in bytes	Description
wHIDDescLength	2	The length of the complete HID descriptor (in Bytes).
bcdVersion	2	The version number, in binary coded decimal (BCD) format.

Member	Size in bytes	Description
wReportDescLength	2	The length of the Report descriptor (in Bytes).
wReportDescRegister	2	The register index containing the Report descriptor.
wInputRegister	2	The register number to read the input report (in unsigned Bytes).
wMaxInputLength	2	The length of the largest input report to be read from the input register.
wOutputRegister	2	The register number to send the output (in unsigned Bytes).
wMaxOutputLength	2	The length of the largest output report to be sent.
wCommandRegister	2	The register number to send command requests (in unsigned Bytes).
wDataRegister	2	The register number to exchange data with command requests (in unsigned Bytes).
wVendorID	2	USB-IF assigned Vendor ID.
wDeviceID	2	Device ID.
wVersionID	2	Firmware version number.

Required Device Attributes

The following HID properties must be provided in the device attributes. The reporting of these device attributes is bus-specific. Consult the HID-specific guidance for your choice of bus.

Expand table

Member	Description	USB	I^2C
wVendorID	Vendor ID	idVendor in USB Device Descriptor	wVendorID in I^2C HID Descriptor (see the preceding table).
wProduct	Product ID	idProduct in USB Device Descriptor	wDeviceID in I^2C HID Descriptor (see the preceding table).
wVersionID	Firmware version number	bcdDevice in USB Device Descriptor	wVersionID I^2C HID Descriptor (see the preceding table).

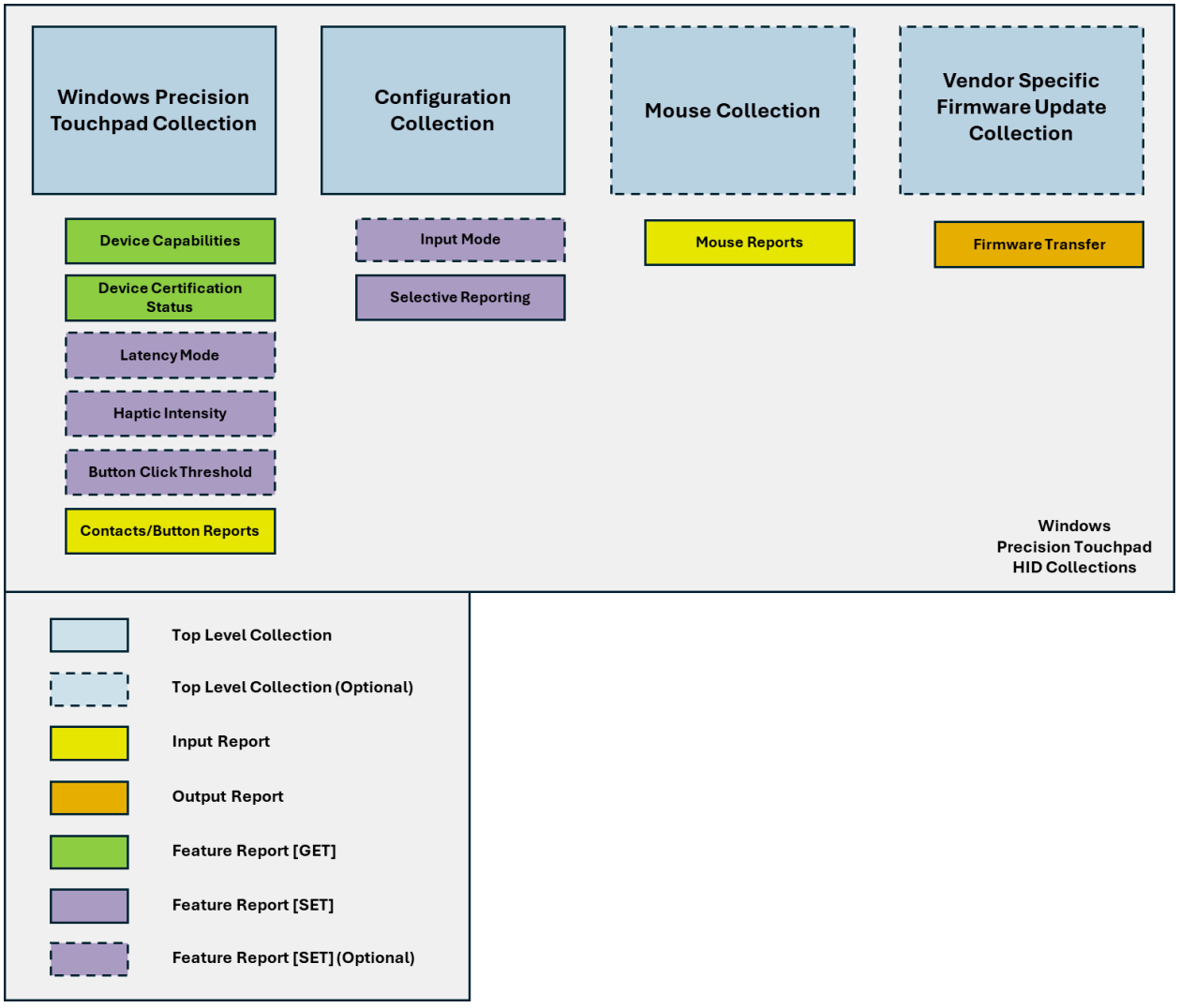
Required HID Top-Level Collections (touchpad-required-hid-top-level-collections)

Article • 03/18/2021

This section discusses the required HID top-level collections that are used for precision touchpad reporting in Windows 10 and later operating systems.

A Windows Precision Touchpad device should expose two mandatory top-level collections: A Windows Precision Touchpad collection, and a Configuration collection. Optional (but recommended) collections for firmware updates and basic mouse mode support may also be implemented.

The following diagram shows the HID collections for a Windows Precision Touchpad device.



A sample descriptor (showing top-level collections) is provided in the Sample Report Descriptors topic.

The following topics provide more details about the HID top-level collections.

In this section

Expand table

Topic	Description
Mouse Collection	This topic discusses the mouse collection of a Windows Precision Touchpad device, and explains how the collection provides HID-compliant mouse reporting to the Windows host.
Configuration Collection	This topic discusses the role played by the configuration collection of a Windows Precision Touchpad device, in Windows 10 and later operating systems.
Windows Precision Touchpad Collection	This topic discusses the top-level collection of a Windows Precision Touchpad, and explains how the collection provides HID-compliant touchpad reporting to the Windows host.
Buttons, Report Level Usages	This topic discusses the report level usages for buttons, within the context of the Windows Precision Touchpad Collection.
Firmware Update Collection (Optional)	A Windows Precision Touchpad device can use the HID protocol to provide a vendor-specific top-level collection for performing device firmware and vendor configuration updates.

Sample Report Descriptors (touchpad-sample-report-descriptors)

Article • 01/29/2024

This topic presents a sample report descriptor and sample descriptor excerpts for a Windows Precision Touchpad device.

> **Note**
>
> Although vendor specific usages may be used at any level of the report, they must be used in a way that could be implemented equally across all reporting methods, namely, serial, parallel, and hybrid. Not adhering to this guidance may not result in an HLK failure but could result in serious compatibility issues.

Sample Report Descriptor

All the values and parameters specified in the following sample descriptor are mandatory.

```
//TOUCH PAD input TLC
    0x05, 0x0d,                         // USAGE_PAGE (Digitizers)
    0x09, 0x05,                         // USAGE (Touch Pad)
    0xa1, 0x01,                         // COLLECTION (Application)
    0x85, REPORTID_TOUCHPAD,            //   REPORT_ID (Touch pad)
    0x09, 0x22,                         //   USAGE (Finger)
    0xa1, 0x02,                         //   COLLECTION (Logical)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //       LOGICAL_MAXIMUM (1)
    0x09, 0x47,                         //       USAGE (Confidence)
    0x09, 0x42,                         //       USAGE (Tip switch)
    0x95, 0x02,                         //       REPORT_COUNT (2)
    0x75, 0x01,                         //       REPORT_SIZE (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x75, 0x02,                         //       REPORT_SIZE (2)
    0x25, 0x02,                         //       LOGICAL_MAXIMUM (2)
    0x09, 0x51,                         //       USAGE (Contact Identifier)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x75, 0x01,                         //       REPORT_SIZE (1)
    0x95, 0x04,                         //       REPORT_COUNT (4)
    0x81, 0x03,                         //       INPUT (Cnst,Var,Abs)
    0x05, 0x01,                         //       USAGE_PAGE (Generic Desk..
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
```

```
    0x26, 0xff, 0x0f,                   //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                         //       REPORT_SIZE (16)
    0x55, 0x0e,                         //       UNIT_EXPONENT (-2)
    0x65, 0x13,                         //       UNIT(Inch,EngLinear)
    0x09, 0x30,                         //       USAGE (X)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x46, 0x90, 0x01,                   //       PHYSICAL_MAXIMUM (400)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x46, 0x13, 0x01,                   //       PHYSICAL_MAXIMUM (275)
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0xc0,                               //    END_COLLECTION
    0x55, 0x0C,                         //    UNIT_EXPONENT (-4)
    0x66, 0x01, 0x10,                   //    UNIT (Seconds)
    0x47, 0xff, 0xff, 0x00, 0x00,      //     PHYSICAL_MAXIMUM (65535)
    0x27, 0xff, 0xff, 0x00, 0x00,         //  LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                           //  REPORT_SIZE (16)
    0x95, 0x01,                           //  REPORT_COUNT (1)
    0x05, 0x0d,                         //    USAGE_PAGE (Digitizers)
    0x09, 0x56,                         //    USAGE (Scan Time)
    0x81, 0x02,                           //  INPUT (Data,Var,Abs)
    0x09, 0x54,                         //    USAGE (Contact count)
    0x25, 0x7f,                           //  LOGICAL_MAXIMUM (127)
    0x95, 0x01,                         //    REPORT_COUNT (1)
    0x75, 0x08,                         //    REPORT_SIZE (8)
    0x81, 0x02,                         //    INPUT (Data,Var,Abs)
    0x05, 0x09,                         //    USAGE_PAGE (Button)
    0x09, 0x01,                         //    USAGE_(Button 1)
    0x09, 0x02,                         //    USAGE_(Button 2)
    0x09, 0x03,                         //    USAGE_(Button 3)
    0x25, 0x01,                         //    LOGICAL_MAXIMUM (1)
    0x75, 0x01,                         //    REPORT_SIZE (1)
    0x95, 0x03,                         //    REPORT_COUNT (3)
    0x81, 0x02,                         //    INPUT (Data,Var,Abs)
    0x95, 0x05,                           //   REPORT_COUNT (5)
    0x81, 0x03,                         //    INPUT (Cnst,Var,Abs)
    0x05, 0x0d,                         //    USAGE_PAGE (Digitizer)
    0x85, REPORTID_MAX_COUNT,             //   REPORT_ID (Feature)
    0x09, 0x55,                         //    USAGE (Contact Count Maximum)
    0x09, 0x59,                         //    USAGE (Pad TYpe)
    0x75, 0x04,                         //    REPORT_SIZE (4)
    0x95, 0x02,                         //    REPORT_COUNT (2)
    0x25, 0x0f,                         //    LOGICAL_MAXIMUM (15)
    0xb1, 0x02,                         //    FEATURE (Data,Var,Abs)
    0x06, 0x00, 0xff,                   //    USAGE_PAGE (Vendor Defined)
    0x85, REPORTID_PTPHQA,                //    REPORT_ID (PTPHQA)
    0x09, 0xC5,                         //    USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //    LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //    LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //    REPORT_SIZE (8)
    0x96, 0x00, 0x01,                   //    REPORT_COUNT (0x100 (256))
    0xb1, 0x02,                         //    FEATURE (Data,Var,Abs)
    0xc0,                               // END_COLLECTION
    //CONFIG TLC
```

```
    0x05, 0x0d,                         //    USAGE_PAGE (Digitizer)
    0x09, 0x0E,                         //    USAGE (Configuration)
    0xa1, 0x01,                         //   COLLECTION (Application)
    0x85, REPORTID_FEATURE,             //   REPORT_ID (Feature)
    0x09, 0x22,                         //   USAGE (Finger)
    0xa1, 0x02,                         //   COLLECTION (logical)
    0x09, 0x52,                         //    USAGE (Input Mode)
    0x15, 0x00,                         //    LOGICAL_MINIMUM (0)
    0x25, 0x0a,                         //    LOGICAL_MAXIMUM (10)
    0x75, 0x08,                         //    REPORT_SIZE (8)
    0x95, 0x01,                         //    REPORT_COUNT (1)
    0xb1, 0x02,                         //    FEATURE (Data,Var,Abs
    0xc0,                               //   END_COLLECTION
    0x09, 0x22,                         //   USAGE (Finger)
    0xa1, 0x00,                         //   COLLECTION (physical)
    0x85, REPORTID_FUNCTION_SWITCH,     //     REPORT_ID (Feature)
    0x09, 0x57,                         //     USAGE(Surface switch)
    0x09, 0x58,                         //     USAGE(Button switch)
    0x75, 0x01,                         //     REPORT_SIZE (1)
    0x95, 0x02,                         //     REPORT_COUNT (2)
    0x25, 0x01,                         //     LOGICAL_MAXIMUM (1)
    0xb1, 0x02,                         //     FEATURE (Data,Var,Abs)
    0x95, 0x06,                         //     REPORT_COUNT (6)
    0xb1, 0x03,                         //     FEATURE (Cnst,Var,Abs)
    0xc0,                               //   END_COLLECTION
    0xc0,                               // END_COLLECTION
    //MOUSE TLC
    0x05, 0x01,                         // USAGE_PAGE (Generic Desktop)
    0x09, 0x02,                         // USAGE (Mouse)
    0xa1, 0x01,                         // COLLECTION (Application)
    0x85, REPORTID_MOUSE,               //   REPORT_ID (Mouse)
    0x09, 0x01,                         //   USAGE (Pointer)
    0xa1, 0x00,                         //   COLLECTION (Physical)
    0x05, 0x09,                         //     USAGE_PAGE (Button)
    0x19, 0x01,                         //     USAGE_MINIMUM (Button 1)
    0x29, 0x02,                         //     USAGE_MAXIMUM (Button 2)
    0x25, 0x01,                         //     LOGICAL_MAXIMUM (1)
    0x75, 0x01,                         //     REPORT_SIZE (1)
    0x95, 0x02,                         //     REPORT_COUNT (2)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x95, 0x06,                         //     REPORT_COUNT (6)
    0x81, 0x03,                         //     INPUT (Cnst,Var,Abs)
    0x05, 0x01,                         //     USAGE_PAGE (Generic Desktop)
    0x09, 0x30,                         //     USAGE (X)
    0x09, 0x31,                         //     USAGE (Y)
    0x75, 0x10,                         //     REPORT_SIZE (16)
    0x95, 0x02,                         //     REPORT_COUNT (2)
    0x25, 0x0a,                          //    LOGICAL_MAXIMUM (10)
    0x81, 0x06,                         //     INPUT (Data,Var,Rel)
    0xc0,                               //   END_COLLECTION
    0xc0,                                //END_COLLECTION
```

Sample Report Descriptor Excerpt - (Latency Mode Feature Report)

The following excerpt is from the descriptor for the Windows Precision Touchpad top-level collection. This descriptor excerpt is for the optional (but highly recommended) support of the latency mode feature report.

```
    0x05, 0x0d,                         //    USAGE_PAGE (Digitizer)
    0x85, REPORTID_LATENCY,             //    REPORT_ID (Latency)
    0x09, 0x60,                         //    USAGE (Latency Mode)
    0x75, 0x01,                         //    REPORT_SIZE (1)
    0x95, 0x01,                         //    REPORT_COUNT (1)
    0x15, 0x00,                         //    LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //    LOGICAL_MAXIMUM (1)
    0xb1, 0x02,                         //    FEATURE (Data,Var,Abs)
    0x95, 0x07,                         //    REPORT_COUNT (7)
    0xb1, 0x03,                         //    FEATURE (Cnst,Var,Abs)
```

Sample Report Descriptor Excerpt - (Haptic Intensity Feature Report)

The following excerpt is from the descriptor for the Windows Precision Touchpad top-level collection for the optional support of the haptic intensity feature report.

```
    0x05, 0x0E,                       //   Usage Page (Haptics)
    0x09, 0x01,                       //   Usage (Simple Haptics Controller)
    0xA1, 0x02,                       //   Collection (Logical)
    0x09, 0x23,                       //     Usage (Intensity)
    0x85, CONFIG_PTP_HAPTICS_ID,      //     Report ID
(CONFIG_PTP_HAPTICS_ID)
    0x15, 0x00,                       //     Logical Minimum (0)
    0x25, 0x64,                       //     Logical Maximum (100)
    0x75, 0x08,                       //     Report Size (8)
    0x95, 0x01,                       //     Report Count (1)
    0xB1, 0x02,                       //     Feature (Data,Var,Abs)
    0xC0,                             //   End Collection ()
```

Sample Report Descriptor Excerpt - (Button Press Threshold Feature Report)

The following excerpt is from the descriptor for the Windows Precision Touchpad top-level collection for the optional support of the button press threshold feature report.

```
    0x05, 0x0D,                         //   UsagePage(Digitizers[0x000D])
    0x85, REPORTID_BUTTONPRESSTHR,      //   ReportId(REPORTID_BUTTONPRESSTHR)
    0x09, 0xB0,                         //   UsageId(Button Press
Threshold[0x00B0])
    0x35, 0x6E,                         //   PhysicalMinimum(110)
    0x46, 0xBE, 0x00,                   //   PhysicalMaximum(190)
    0x66, 0x01, 0x01,                   //   Unit('gram', SiLinear, Gram:1)
    0x15, 0x01,                         //   LogicalMinimum(1)
    0x25, 0x03,                         //   LogicalMaximum(3)
    0x75, 0x02,                         //   ReportSize(2)
    0xB1, 0x02,                         //   Feature(Data, Variable, Absolute)
    0x75, 0x08,                         //   ReportSize(8)
    0xB1, 0x03,                         //   Feature(Constant, Variable,
Absolute)
```

Sample Report Descriptor Excerpt - (Segmented Device Certification Status Feature Report)

The following excerpt is from the descriptor for the Windows Precision Touchpad top-level collection, for a segmented version of the device certification status feature report. This allows the certification status blob to be split into eight 32-byte segments, versus a single 256-byte segment. The host indicates which segment # it would like returned in a SET FEATURE. In response, the device returns the segment #, along with the associated segment in GET FEATURE.

```
    0x06, 0x00, 0xff,                   //     USAGE_PAGE (Vendor Defined)
    0x85, REPORTID_PTPHQA,              //     REPORT_ID (PTPHQA)
    0x09, 0xC6,                         //     USAGE (Vendor usage for
segment #)
    0x25, 0x08,                         //     LOGICAL_MAXIMUM (8)
    0x75, 0x08,                         //     REPORT_SIZE (8)
    0x95, 0x01,                         //     REPORT_COUNT (1)
    0xb1, 0x02,                         //     FEATURE (Data,Var,Abs)
    0x09, 0xC7,                         //     USAGE (Vendor Usage)
    0x26, 0xff, 0x00,                   //     LOGICAL_MAXIMUM (0xff)
    0x95, 0x20,                         //     REPORT_COUNT (32)
    0xb1, 0x02,                         //     FEATURE (Data,Var,Abs)
```

Sample Report Descriptor Excerpt - (PTP Input Report with Geometry Support)

```
    0x85, REPORTID_TOUCHPAD,            //   REPORT_ID (Touch pad)
    0x09, 0x22,                         //   USAGE (Finger)
    0xa1, 0x02,                         //   COLLECTION (Logical)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //       LOGICAL_MAXIMUM (1)
    0x09, 0x47,                         //       USAGE (Confidence)
    0x09, 0x42,                         //       USAGE (Tip switch)
    0x95, 0x02,                         //       REPORT_COUNT (2)
    0x75, 0x01,                         //       REPORT_SIZE (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x75, 0x02,                         //       REPORT_SIZE (2)
    0x25, 0x02,                         //       LOGICAL_MAXIMUM (2)
    0x09, 0x51,                         //       USAGE (Contact Identifier)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x75, 0x01,                         //       REPORT_SIZE (1)
    0x95, 0x04,                         //       REPORT_COUNT (4)
    0x81, 0x03,                         //       INPUT (Cnst,Var,Abs)
    0x05, 0x01,                         //       USAGE_PAGE (Generic
Desktop)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x0f,                   //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                         //       REPORT_SIZE (16)
    0x55, 0x0e,                         //       UNIT_EXPONENT (-2)
    0x65, 0x13,                         //       UNIT(Inch,EngLinear)
    0x09, 0x30,                         //       USAGE (X)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x46, 0x90, 0x01,                   //       PHYSICAL_MAXIMUM (400)
    0x95, 0x02,                         //       REPORT_COUNT (2)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x46, 0x13, 0x01,                   //       PHYSICAL_MAXIMUM (275)
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x05, 0x0d,                         //       USAGE_PAGE (Digitizers)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x25, 0x64,                         //       LOGICAL_MAXIMUM (100)
    0x95, 0x02,                         //       REPORT_COUNT (2)
    0x09, 0x48,                         //       USAGE (Width)
    0x09, 0x49,                         //       USAGE (Height)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0xc0,                               //    END_COLLECTION
```

Related topics

Configuration Collection

Mouse Collection

Windows Precision Touchpad Collection

Precision touchpad tuning (touchpad-tuning-guidelines)

Article • 01/29/2024

There are several approaches an OEM can take to adjust the default touchpad experience on their Windows-based device. The default values meet the needs of most users, so they should only be adjusted if they don't provide the specific experience that the OEM wants (and in the case of HorizontalOffset / SpaceBarOffset, which should both always be set).

> **Note**
>
> Be aware that serious problems can occur if the registry is modified incorrectly. Ensure these steps are followed carefully. For additional protection, back up the registry before modification. See **How to back up and restore the registry in Windows** .

The tunable registry values described below are all located in the touchpad device's HKR hardware key:

- Introduction to registry keys for drivers
- Device Driver INF AddReg Directive
- Opening a Device's Hardware Key

HKEY_LOCAL_MACHINE (HKLM)

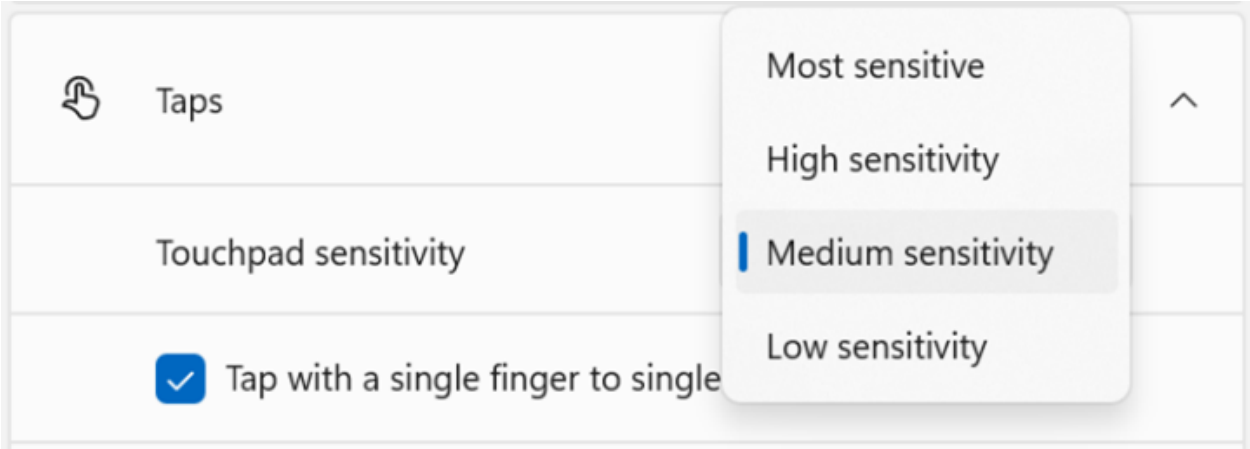
On versions of Windows prior to version 1903, only the global location was supported. The default global location is:

`HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\PrecisionTouchPad`

On Windows, version 1903 and later, each of the registry values in this guide are checked for under the HKR key first, and then under the `HKEY_LOCAL_MACHINE` key next if not found. This enables the ability to override defaults in HKLM by using the HKR location on a per-value basis.

Accidental Activation Prevention

AAP is a feature of Precision Touchpads that suppresses certain actions like mouse moves and taps for a short time after keyboarding events. The specific timeouts depend on the type of action and the region of the touchpad receiving input, as well as the user setting called Touchpad sensitivity found in Touchpad settings. More sensitive means less time is spent suppressing input after keyboarding events. Maximum sensitivity means AAP timeouts are zero and AAP is not in effect.



Tunable parameters

Windows defines two classes of "Curtain" regions on the touchpad to help prevent accidental input when users are resting their hands on the keyboard or touchpad during normal use. This splits the touchpad into three regions:

- **High sensitivity region:** input suppression is minimal in this region
- **Curtain region:** input may be suppressed for some time in this region
- **Supercurtain region:** certain kinds of input like taps are almost always suppressed when the input originates with this region

Even when specifying curtain sizes explicitly, it is strongly recommended to provide the physical touchpad positioning information via `SpaceBarOffset` and `HorizontalOffset`. This is necessary for Windows to make decisions around a consistent and predictable user experience when switching between keyboarding and pointing. For more information, see the Windows Precision Touchpad Device Integration guide.

Curtains

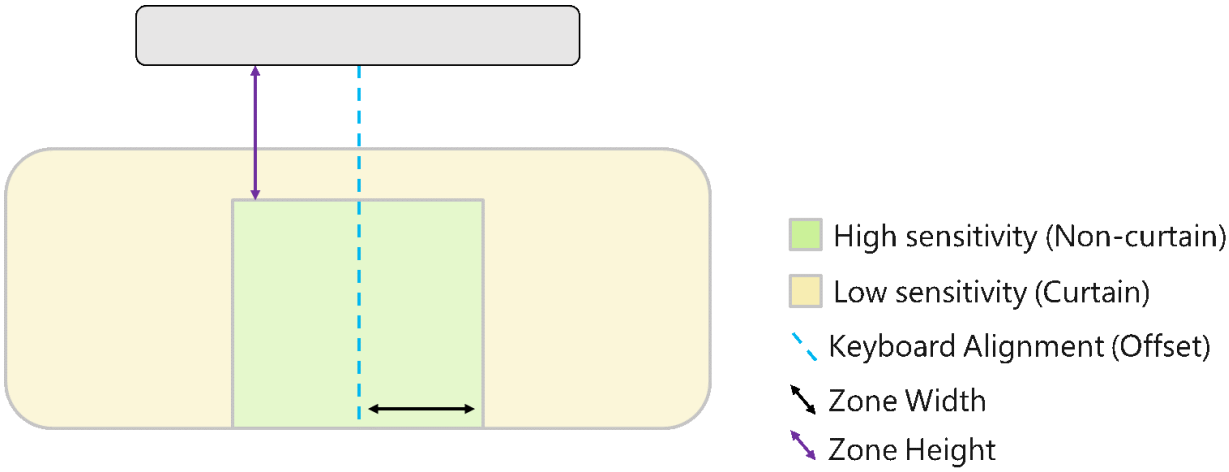
If the three values `CurtainLeft`, `CurtainRight`, and `CurtainTop` are found, Curtains will be constructed using these as distances from the respective edges. If all three of these values are not found, the legacy configuration (see Legacy Curtains below) will be used next, followed by defaults.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
CurtainTop	DWORD	Himetric	0+	0	Windows 10, version 1903+
CurtainLeft	DWORD	Himetric	0+	0	Windows 10, version 1903+
CurtainRight	DWORD	Himetric	0+	0	Windows 10, version 1903+
CurtainBottom (Optional)	DWORD	Himetric	0+	0	Windows 10, version 1903+

Curtains (Implicit)

The legacy algorithm for configuring curtains changes curtain sizes depending on the vertical location of touchpad event, and isn't as flexible for customization. Going forward, it is recommended that only `SpaceBarOffset` and `HorizontalOffset` values be set, and curtains be otherwise configured through `CurtainTop`, etc. described above.



The high sensitivity region is intended to stay responsive even during or immediately after keybaord use.

The width of the region on either side of the center line (Zone Width in the diagram above) is defined as:

SpacebarOffset + AAPNonCurtainBaseWidth

Note that this means the total width of the region is twice this value. `SpacebarOffset` is taken into account for the region, since proximity to the keyboard increases risk of palm contact.

The height of the region is defined in relation to the space bar and stretches to the bottom of the touchpad. Distance to the top of the zone is defined as:

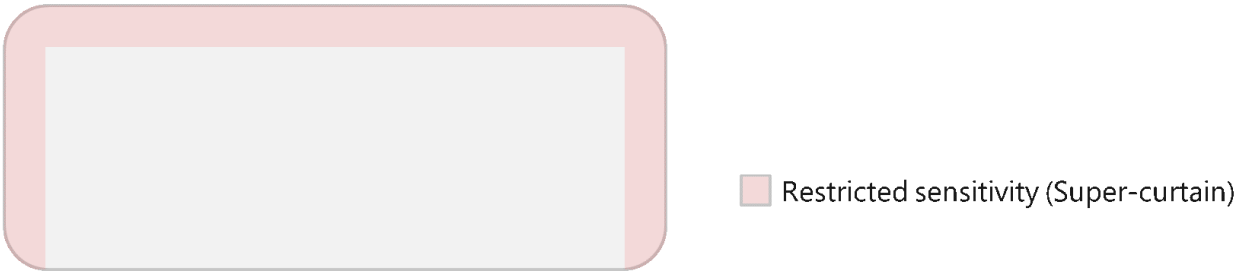
AAPNonCurtainTop - SpaceBarOffset

Expand table

Registry Value	Type	Units	Range	Default	Availablity
SpaceBarOffset	DWORD	Himetric	0+	1000	Windows 8.1 +
HorizontalOffset	DWORD	Himetric	Any	0	Windows 8.1 - Windows 10, version 1511 for 0+ range, Windows 10, version 1511+ for any range
HorizontalOffsetIsNeg	DWORD	Himetric	0-1	0	Windows 8.1 +, deprecated in Windows 10, version 1511
AAPNonCurtainTop	DWORD	Himetric	0+	1500	Windows 10, version 1607+
AAPNonCurtainBaseWidth	DWORD	Himetric	0+	1500	Windows 10, version 1607+

Super Curtains

The restricted sensitivity zone is the lowest sensitivity region of the touchpad. It is designed for the region of the touchpad where users would not leave fingers during intentional use. Note these are very small by default and should only be widened for very large devices.



The top, left, and right zones' sizes are defined independently as a distance from their respective edge. They use the registry keys shown in the following table.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
SuperCurtainTop	DWORD	Himetric	0+	100	Windows 10, version 1511+
SuperCurtainLeft	DWORD	Himetric	0+	100	Windows 10, version 1511+

Registry Value	Type	Units	Range	Default	Availablity
SuperCurtainRight	DWORD	Himetric	0+	100	Windows 10, version 1511+
SuperCurtainBottom	DWORD	Himetric	0+	100	Windows 10, version 1903+

Right-click zone

The following registry values allow the configuration of the right click zone. The height is expressed by `RightClickZoneHeight` as a percentage of the height of the entire touchpad measured from the bottom edge. `RightClickZoneWidth` is also a percentage of the width, measured from the right edge in right-handed, and the left edge in left-handed configurations when the swap mouse buttons option is enabled.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
RightClickZoneHeight	DWORD	Percent	1-100	25	Windows 10, version 1507+
RightClickZoneWidth	DWORD	Percent	1-100	50	Windows 10, version 1507+

Suppressing Accidental Activation Protection

Setting the following key is equivalent to the user choosing "Most sensitive", which disables AAP. When this key is set, the Touchpad sensitivity setting in Windows Settings (shown above) has no effect.

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\AAPDisabled`

Expand table

Registry Value	Type	Units	Range	Default	Availablity
AAPDisabled	DWORD	n/a	0-1	0	Windows 10, version 1511+

Detecting keyboard mouse collection as a non-mouse

Windows provides a registry setting that lets you exempt specific devices. This can be found in:

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\IgnoredExternalMice`

Structuring this key is similar to forcing a legacy touchpad to be enabled. See legacy touchpad forced detection for more details.

HKEY_CURRENT_USER (HKCU)

OEMs are able to change the user's touchpad settings by modifying the below HKCU registry values. All values are under the registry key `HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad`. To change the defaults used when a new user profile is created, modifications should be made to the registry hive located at `%SystemDrive%\Users\Default\NTUSER.DAT`.

Leave touchpad on when a mouse is connected

This setting specifies whether the touchpad is allowed to remain active when an external mouse is present. When inactive, any input produced by the touchpad is ignored.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
LeaveOnWithMouse	DWORD	n/a	0-1	1	Windows 10, version 1511+

Cursor speed

This setting specifies the rate at which the mouse motion produced by the touchpad moves the cursor. Valid values are 1-20, inclusive. The user interface for this setting limits the range to 2-20 with a step of 2.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
CursorSpeed	DWORD	n/a	1-20	10	Windows 10, version 1511+

Click sensitivity

This setting specifies the relative sensitivity of the touchpad's haptic click detection, if supported. Valid values are 0-100, inclusive. The user interface for this setting limits the

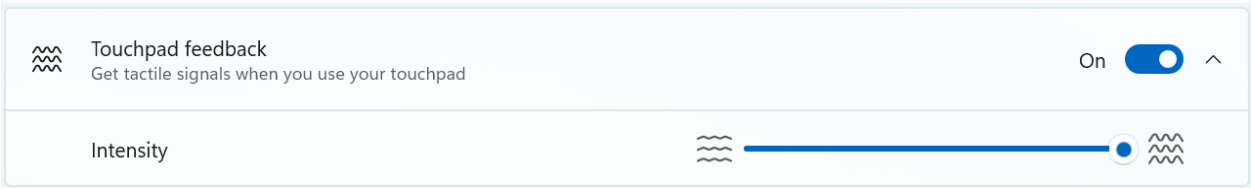
range to 0-100 with a step of 50. This is the user-facing setting for the "Button Press Threshold" feature report.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
ClickForceSensitivity	DWORD	Percent	0-100	50	Windows 11, build 26027+

"Touchpad feedback" section

Windows provides the ability to modify settings in the **Touchpad feedback** section of Touchpad settings.



Feedback enabled

This setting specifies whether haptic feedback is enabled on touchpads where it is supported.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
FeedbackEnabled	DWORD	n/a	0-1	1	Windows 11

> **Note**
>
> To ensure a great user experience, haptic feedback should be enabled by default. Therefore this value should remain set to `1`.

Intensity

This setting specifies the relative intensity of the touchpad's haptic feedback, if supported. Valid values are 0-100, inclusive. The user interface for this setting limits the range to 0-100 with a step of 25. This is the user-facing setting for the "Haptic Intensity" feature report.

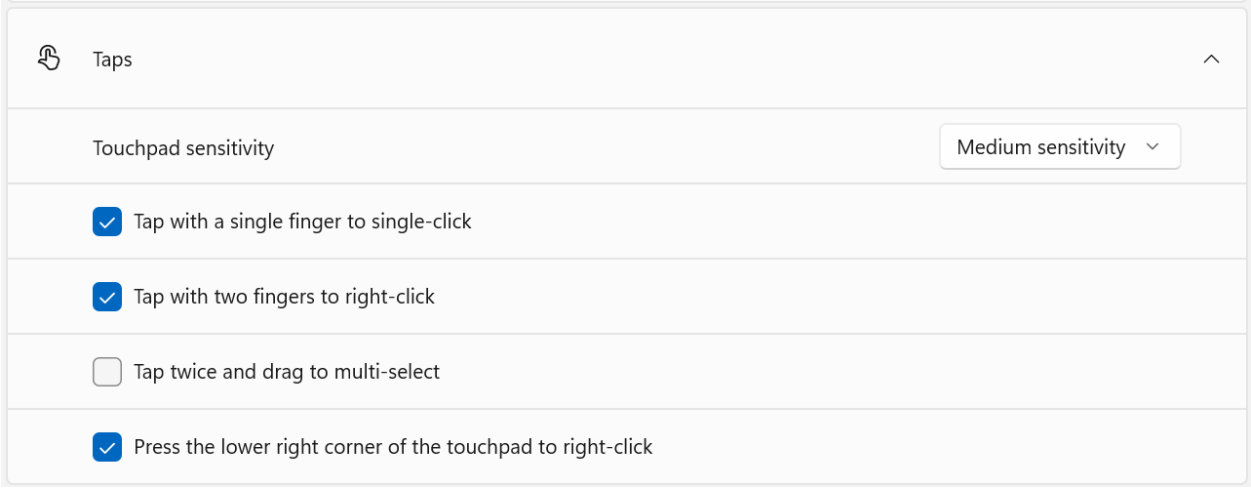
> **Note**
>
> To ensure a great user experience, haptic feedback should be enabled by default. Therefore this value should be set to a non-zero value.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
FeedbackIntensity	DWORD	Percent	0-100	50	Windows 11

"Taps" section

Windows provides the ability to modify settings in the **Taps** section of Touchpad settings.


Taps
Touchpad sensitivity
Medium sensitivity
Tap with a single finger to single-click
Tap with two fingers to right-click
Tap twice and drag to multi-select
Press the lower right corner of the touchpad to right-click

Touchpad sensitivity

This setting specifies the touchpad's sensitivity level. The more sensitive the touchpad, the less suppression of mouse input generation occurs after keyboard activity. Lower values correspond to higher sensitivity (meaning 0 indicates highest sensitivity).

Expand table

Registry Value	Type	Units	Range	Default	Availablity
AAPThreshold	DWORD	n/a	0-4	2	Windows 10, version 1511+

Tap with a single finger to single-click

This setting specifies whether single-finger taps are enabled.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
TapsEnabled	DWORD	n/a	0-1	1	Windows 10, version 1511+

Tap with two fingers to right-click

This setting specifies whether two-finger taps are enabled.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
TwoFingerTapEnabled	DWORD	n/a	0-1	1	Windows 10, version 1511+

Tap twice and drag to multi-select

This setting specifies whether the tap-and-drag gesture is enabled.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
TapAndDrag	DWORD	n/a	0-1	1	Windows 10, version 1511+

Press the lower right corner of the touchpad to right-click

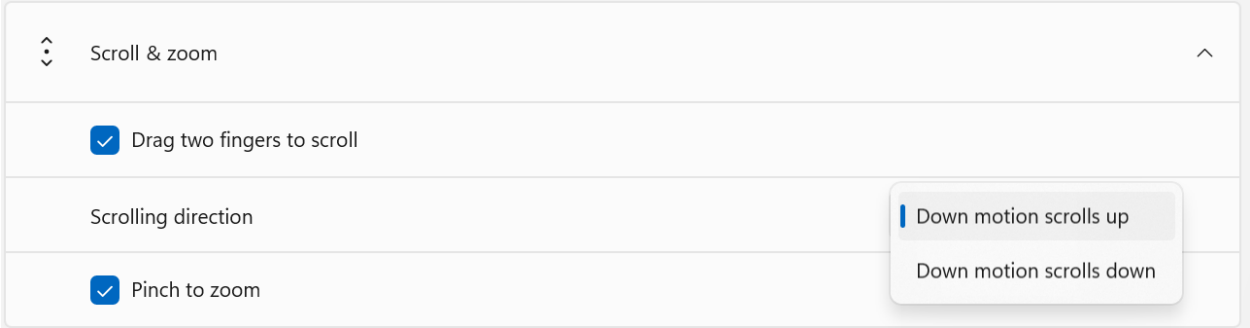
This setting specifies whether pressing the touchpad in its bottom-right corner results in a right-click being produced instead of a left click. If the user has swapped their left and right mouse buttons (`GetSystemMetrics(SM_SWAPBUTTON)` is true), the right-click zone is mirrored horizontally to the bottom-left corner of the touchpad.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
RightClickZoneEnabled	DWORD	n/a	0-1	1	Windows 10, version 1511+

"Scroll & zoom" section

Windows provides the ability to modify settings with the **Scroll & zoom** section of Touchpad settings.



> **Note**
>
> There is currently no way to modify the scrolling or zooming speed of a touchpad.

Drag two fingers to scroll

This setting specifies whether two-finger panning is enabled.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
PanEnabled	DWORD	n/a	0-1	1	Windows 10, version 1511+

Scroll direction

This setting specifies whether the direction content scrolls via two-finger panning is reversed. In the default state, upwards motion of contacts on the touchpad results in content scrolling downward and leftwards motion of contacts results on content scrolling rightwards.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
ScrollDirection	DWORD	n/a	0-1	0	Windows 10, version 1511+

Pinch to zoom

This setting specifies whether two-finger zooming is enabled.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
ZoomEnabled	DWORD	n/a	0-1	1	Windows 10, version 1511+

Settings with no user interface

The following settings are supported by Windows but are not exposed to the user in the Settings application.

Mouse acceleration

This setting specifies whether the mouse motion produced by the touchpad honors the user's mouse acceleration setting as returned by `SystemParametersInfo(SPI_GETMOUSE)`. If 0, the mouse motion will always have acceleration applied.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
HonorMouseAccelSetting	DWORD	n/a	0-1	0	Windows 10, version 1511+

Right click zone width

This setting specifies the relative width of the touchpad's right-click zone. Valid values are 0-100, inclusive. If non-zero, it overrides the device's configuration.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
RightClickZoneWidth	DWORD	Percent	0-100	0	Windows 11, build 26027+

Right click zone height

This setting specifies the relative height of the touchpad of the touchpad's right-click zone. Valid values are 0-100, inclusive. If non-zero, it overrides the device's configuration.

Expand table

Registry Value	Type	Units	Range	Default	Availablity
RightClickZoneHeight	DWORD	Percent	0-100	0	Windows 11, build 26027+

Dynamically querying and modifying settings

As of Windows 11, build 26027, the user's touchpad settings can be queried and modified dynamically via the `SystemParametersInfo` API. For example:

```
C++

// Get the user's touchpad settings.
TOUCHPAD_PARAMETERS params = {};
params.versionNumber = TOUCHPAD_PARAMETERS_LATEST_VERSION;
RETURN_IF_WIN32_BOOL_FALSE(
    ::SystemParametersInfo(SPI_GETTOUCHPADPARAMETERS, sizeof(params),
&params, 0 /*fWinIni*/));

// Override the touchpad's right-click zone to have a 50% width and 33%
height.
// NOTE: changing user settings should only be performed in response to
explicit user intent.
params.rightClickZoneWidth = 50;
params.rightClickZoneHeight = 33;

RETURN_IF_WIN32_BOOL_FALSE(
    ::SystemParametersInfo(SPI_SETTOUCHPADPARAMETERS, sizeof(params),
&params, SPIF_UPDATEINIFILE | SPIF_SENDCHANGE));
```

As with any call to `SystemParametersInfo`, the process must be running under the user's account in order for the settings to be correctly queried or modified.

Experience Customization (touchpad-experience-customization)

Article • 01/18/2019

This topic provides design and testing guidelines for experience customization for Windows Precision Touchpad devices on Windows 10 and later operating systems.

Starting with Windows 10, it is possible to use customization filter drivers to provide device-specific experiences for a touchpad, beyond the core interactions supported in Windows.

Changing the behaviors of the device can lead to unexpected or damaged user experiences, so the following guidance should be taken as strong recommendations for any 3rd party who would like to use a filter driver to provide a differentiated user experience.

User Experience Guidelines

All customizations should be built to produce an additive experience for users to improve the user experience in ways beyond what are already present without the driver. The following guidelines outline the behaviors that should be avoided to ensure a good user experience:

- Product owners should not replace core Windows behaviors. Precision Touchpad devices provide a standard core set of interactions, and user expectations will be that these are available and predictable. Windows core gestures should be consistent between machines and driver versions. A consistent input experience is critical to inspire user confidence and modifying the core set of Precision Touchpad interactions will negatively impact confidence.

- Product owners should not build new interactions that duplicate existing experiences. An excess of gestures that perform the same interaction can confuse users and make accidental gestures more likely. The following table presents the core Windows experiences on a touchpad device, showing how the Tap, Slide and Pinch & Spread actions are interpreted.

Expand table

Contact type	Tap	Slide	Pinch / Spread
One finger	Click	One-finger mousing	N/A

Contact type	Tap	Slide	Pinch / Spread
Two fingers	Secondary click	Panning	Zoom
Three fingers	Windows Shell interaction	Windows Shell interaction	N/A
Four fingers	Windows Shell interaction	Windows Shell interaction	N/A

- Mousing is the most basic and common touchpad interaction and should be protected for users. Avoid one finger gestures that would easily interfere with mousing such as drawing shapes, like characters or circles, with a single finger.
- The manipulation recognition component is highly sensitive to changes in data. Avoid two, three, and four-finger interactions that may be detected immediately before, immediately after, or concurrent with inbox gestures, like for example, drawing a check mark with three fingers. Be sure to perform thorough usability testing of gestures in this space, to ensure that core gestures are still responsive and accurate.
- If a custom gesture requires a constant stream of data, do not send data simultaneously to the Windows gesture processor as well. Doing so may result in interactions firing from the operating system during custom gesture detection. To ensure that this does not affect responsiveness of the system, avoid gestures that need to consume data for noticeable periods of time.
- Ensure that users are deliberately performing a custom gesture before cutting off the flow of data to Windows. Performing a custom four-finger slide gesture immediately when a fifth finger is present, for instance, can lead to a damaged four-finger gesture experience.
- If a gesture implements discretely, such as a gesture that triggers on tap or zone entry, do not implement the gesture during a continuous gesture created either by the system or the customization component. Doing so will result in an action triggering, while another interaction is still processing.
- All gestures must be user configurable. At a minimum, it should be possible to switch the detection of all interactions on or off. Gesture settings should be made available in the system settings application. There are future plans to provide instructions for implementing feature settings.

Guidance for Testing

Customization drivers are likely to cause failures for devices during compatibility tests. These tests are intended to test hardware capabilities, and the filter driver may interrupt the expected data flow to the tests. However, if you plan to ship your systems with customization components, then these components should be included during testing. So these customized behaviors must be implemented in such a way that they can be disabled during testing, if they would otherwise block the device from passing certification.

Since, as noted above, customized interactions should always have user configurable settings to disable or enable the behaviors, it should be possible for any filter driver to have its customizations disabled to allow testing of the hardware capabilities.

Gesture Design Principles

Windows uses the following design principles in designing gestures. These ensure that gestures provide meaningful additions to user experiences.

Deliberate: Gestures should not be accidentally activated easily. Chose physical gestures that are both easy to invoke intentionally and hard to invoke unintentionally. Gestures should be reversible or terminable wherever appropriate.

Productive: Gestures should help users be more productive. Touchpad devices are primarily productivity devices, and helping users do what they want to do faster, provides value. Gestures that make users productive will keep users on their systems.

Delightful: Gestures should delight users by providing exciting ways to interact with their system that impress users when shown or discovered – gestures that delight users will help bring users to systems.

Intuitive: Gestures should have an intuitive mapping between physical motions and functional behaviors in the system. This makes the gestures easy to learn, and they will feel more natural to use.

Breadth: Gestures should be useful in as many situations as possible. Gestures that require certain Apps or frameworks in order to function, take up valuable space in the touchpad gesture set.

Device Integration (touchpad-device-integration)

Article • 01/24/2024

This topic discusses the considerations and requirements for integrating a Windows Precision Touchpad device into a host system.

Windows Precision Touchpad devices define an experience, and integration of the device has a significant impact on how accurately that experience is implemented. The following sections provide information about the various factors that have to be considered for a successful device integration.

Size

A Windows Precision Touchpad device should have a sensor with minimum dimensions of 32mm x 64mm, as shown in the following diagram. This should be the minimum permissible size that is reported via the physical maximum for X and for Y in the report descriptor.

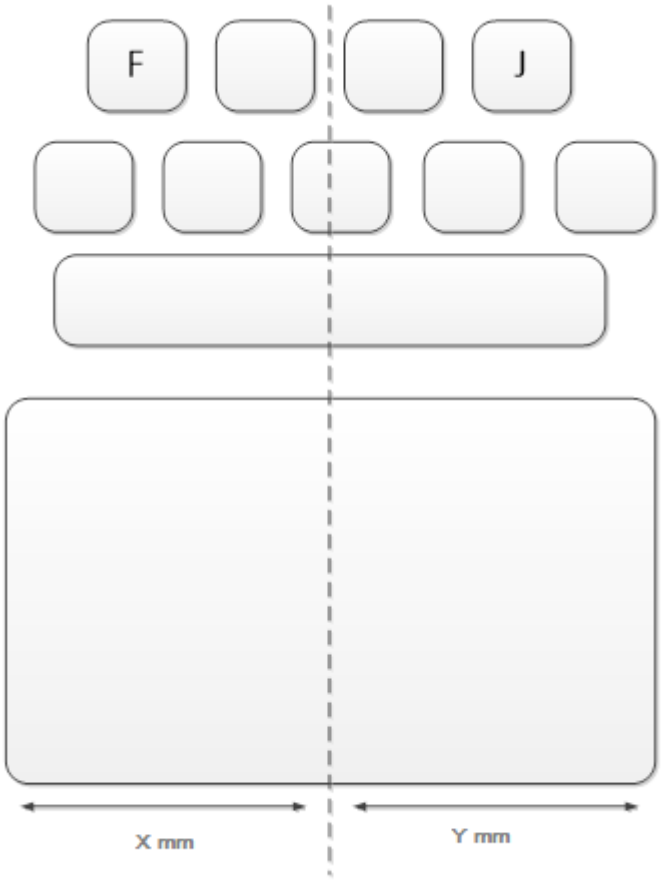
The best Windows Precision Touchpad devices should have the recommended dimensions of approximately 65mm x 105mm, as shown in the following diagram, to allow for more comfortable interactions.

Placement

Windows Precision Touchpad placement is defined by three measurements: horizontal offset, vertical offset and depth offset.

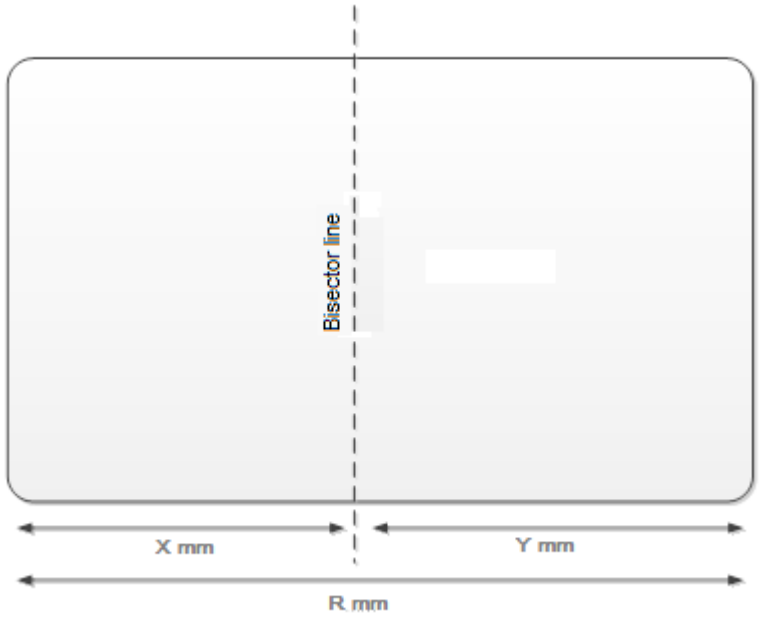
Horizontal Offset

The optimal placement for a Windows Precision Touchpad is to center the device with the line which bisects the 'F' and 'J' keys of the integrated keyboard as shown in the following diagram.



If a Windows Precision Touchpad device cannot be integrated with the optimal zero offset, then the integrator should store the positive or negative offset value (in himetric units) in the registry to allow the host device to compensate.

If a device has an offset, the value to store is calculated by taking the length of the touchpad to the right of the bisecting line (Y) and subtracting the length of the touchpad to the left of the bisecting line (X), such that Y – X = Offset value. If a device has a right offset, this value will be positive, whereas a device with a left offset will result in a negative value. The following diagram shows the 'X' and 'Y' distances referenced in the preceding explanation.



The Windows registry key that stores information about the horizontal offset of the touchpad is:

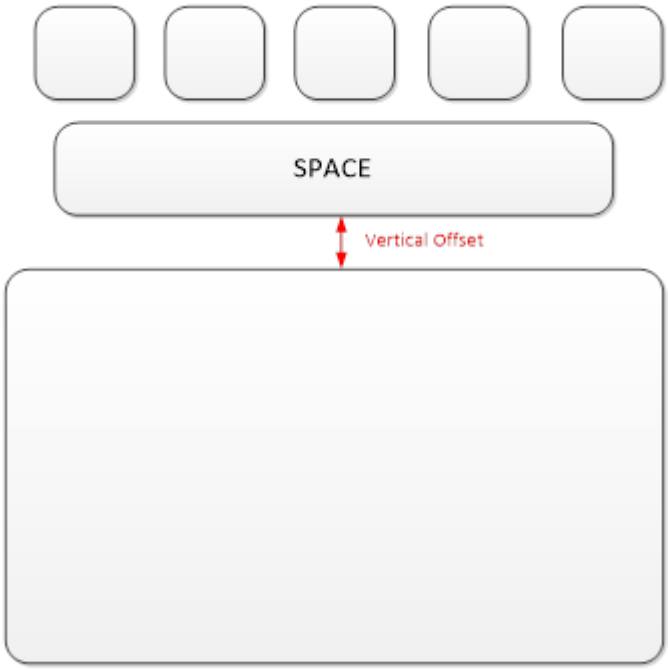
HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\PrecisionTouchPad And here are the relevant variables with their values:

Expand table

Setting	Name	Type	Default value	Value
Horizontal Offset	HorizontalOffset	DWORD	0	Absolute offset distance in himetric units.
Indicate Negative	HorizontalOffsetIsNeg	DWORD	0	0 = Positive Offset 1 = Negative Offset

Vertical Offset

Windows Precision Touchpad devices can be integrated at different vertical offsets from the keyboard spacebar as shown in the following diagram. The integrator should store the positive offset (in himetric units) in the registry, to allow the host to compensate. If a value is not provided, the host should assume a default offset of 14mm.



The Windows registry key that stores information about the vertical offset of the touchpad is:

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\PrecisionTouchPad And here is the relevant variable with its default value:

Expand table

Setting	Name	Type	Default value	Value
Vertical Offset	SpaceBarOffset	DWORD	1000	Offset distance in himetric units.

Note If the touchpad is not below the space bar, but is in fact located above the keyboard, leave the vertical offset at the default value.

Depth Offset

See Palm Deck Integration for more information on integrating the touchpad with the palm deck, including recommendations for depth.

Module Design for Windows HCK Requirements (touchpad-module-design-for-windows-hck-requirements)

Article • 01/24/2024

The Windows Hardware Certification Kit (HCK) requirements for Windows Precision Touchpads are designed to provide a consistent user experience where precision and reliability are at the forefront. These requirements shall influence all aspects of the module, including the sensor, controller IC and associated mechanics.

Sensor design

The design of the sensor in the Windows Precision Touchpad module is essential to ensuring an accurate representation of the user's finger interactions.

Although a specific sensor pitch is not mandated in this implementation guide, it should be understood how a larger sensor pitch can introduce challenges when attempting to meet or exceed specific requirements.

Minimum input separation

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Performance.MinSeparation
- Device.Input.PrecisionTouchpad.Precision.ContactDivergence
- Device.Input.PrecisionTouchpad.Precision.HVInputSeparation
- Device.Input.PrecisionTouchpad.Precision.DiagonalInputSeparation

Ensuring that each unique finger contact is identified and reported is essential for consistent and reliable gesture recognition.

Windows Precision Touchpads shall not alias contacts aligned vertically or horizontally at a minimum separation of 10mm or aligned diagonally at a minimum separation of 13mm irrespective of whether the contacts are stationary, diverging, converging or being interleaved.

Surface and edge contact detection

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Precision.EdgeDetection
- Device.Input.PrecisionTouchpad.Reliability.ContactsReported

Ensuring that contacts are registered and reported as close to the edge of the sensor is essential for consistent and reliable edge gesture recognition.

Windows Precision Touchpads shall detect and report contacts anywhere on the digitizer surface within a maximum of 2mm of the edge of the digitizer surface irrespective of whether the contacts are within, entering or exiting the sensor area.

Controller IC design

The design of the controller IC in the Windows Precision Touchpad module is essential to ensuring an accurate representation of the user's finger interactions.

Position Reporting

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Precision.MotionJitter
- Device.Input.PrecisionTouchpad.Precision.Position
- Device.Input.PrecisionTouchpad.Precision.StationaryJitter

The kinematics of the surface contacts shall be reported as accurately as possible to the host by a Windows Precision Touchpad. If a contact is stationary it shall be reported with stationary coordinates. A moving contact shall have its position accurately reported with respect to the scan time value.

Linearity

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Precision.Linearity

The reporting of subtle movements by the user is an essential part of a precise and responsive user experience; however the lack of deviation and ability to follow the vector of a finger precisely is just as critical.

Windows Precision Touchpads shall maintain linearity within 0.5mm for all contacts reported across edge to edge travel horizontally, vertically, and diagonally. Within 3.5mm of any edge, precision touchpads shall maintain linearity within 1.5mm for all contacts reported.

Figure 1 Linearity

Latency and report rate

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Precision.ActiveTouchdownLatency
- Device.Input.PrecisionTouchpad.Precision.IdleTouchDownLatency
- Device.Input.PrecisionTouchpad.Precision.PanLatency
- Device.Input.PrecisionTouchpad.Performance.ReportRate

User-perceived latency significantly diminishes the experience of a Windows Precision Touchpad and therefore all aspects of the system from end-to-end shall meet or exceed specified latency goals. Providing a minimal input report rate of 125Hz for single contacts and 100Hz for multiple contacts ensures that with the correct scan frequencies, contact down and update latencies of 25ms and 15ms respectively can be achieved.

Reliability

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Reliability.ContactSuppression
- Device.Input.PrecisionTouchpad.Reliability.FalseContacts
- Device.Input.PrecisionTouchpad.Reliability.PowerStates

The most critical aspect of a digitizer system is ensuring that spurious contacts are not reported. Spurious contacts can occur due to noise interference that is introduced to the

system from a variety of sources; the Windows Precision Touchpad controller shall ensure that these are never reported to the host.

A user can make contact with a Windows Precision Touchpad at any time (either intentional or inadvertent) and the controller must ensure that it can boot correctly irrespective of surface contacts or button state and be able to report contacts in accordance with the HCK requirements once all the initial contacts have been removed. Should a Windows Precision Touchpad detect more contacts on the surface than is supported for contact reporting and tracking, it shall report an up for all contacts and buttons, and cease all reporting until all contacts have been removed.

Mechanical design

The design of the mechanics in the Windows Precision Touchpad module is essential to ensuring a consistent user experience.

Button activation force

Related HCK requirements:

- Device.Input.PrecisionTouchpad.Hardware.ClickpadPress
- Device.Input.PrecisionTouchpad.Hardware.PressurePadPress

Irrespective of button type implementation, a button down state shall be reported by a Windows Precision Touchpad when a force that is greater than 150g-180g is applied to the contact area. The best Windows Precision Touchpads shall strive to provide uniform activation across the entire contact area (this is required for pressure-pad implementations), however at a minimum Windows Precision Touchpads shall ensure that applied activation force, as shown in *Figure 2 Activation Force*, results in button down reporting.

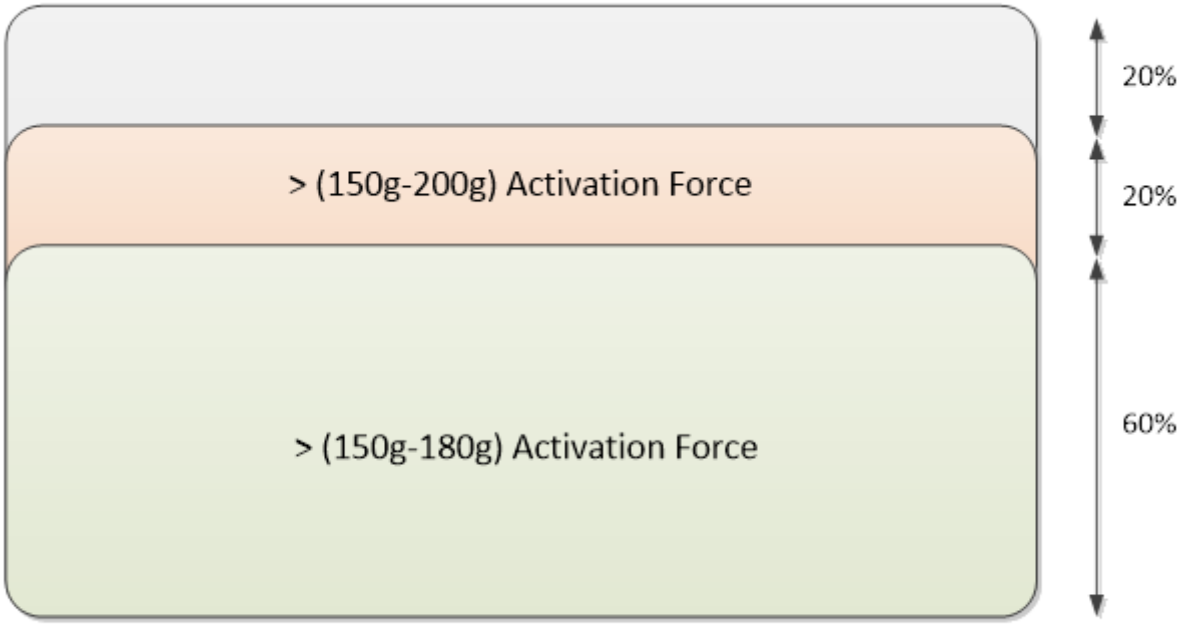


Figure 2 Activation Force

Enable, Disable Toggle Button (touchpad-enable-or-disable-toggle-button)

Article • 01/24/2024

Windows Precision Touchpad devices (or legacy touchpad devices that were configured for enable/disable control in Windows 8.1), can have their enable/disable state toggled via a hardware button, or a keyboard combination.

For convertible devices that implement integrated touchpads on keyboards that can be folded back, a device driver that typically disables keyboard or button features can also disable the precision touchpad. To disable the precision touchpad, the device driver should first query the following registry key:

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\Status\Enabled]

If this value is non-zero, the driver can issue a hotkey combination [**CTRL + WIN + F24**] to toggle the precision touchpad OFF. If the keyboard is no longer folded back, and the value of the registry key is zero, then the driver can toggle it back ON.

Note This registry key is read only, and is not to be modified directly.

So in short, the state of the touchpad will be toggled when the **CTRL + WIN + F24** keyboard combination is reported to the host.

Windows Precision Touchpad Industrial Design Guide (touchpad-industrial-design-guide)

Article • 01/24/2024

This topic provides industrial design guidance for Windows Precision Touchpads. Many of these guidelines are part of the Windows Hardware Certification Kit (HCK) and are identified as such in this topic. Additional industrial design characteristics are provided as suggested guidance and reference for hardware manufacturers. This information applies to Windows 8.1.

In this section

Expand table

Topic	Description
Surface Treatment	The coversheet of a Windows Precision Touchpad delights the user by its feel. The coversheet is manufactured with a high quality coating and machined texture to produce a glass-like smoothness.
Palm Deck Integration	Integration of a precision touchpad with the palm deck impacts the ease of use of the touchpad. Precision touchpads are designed to be natural and intuitive interface for all users. Users should able to easily identify the interaction surface, gesture in a natural and responsive fashion, and seamlessly interact with the touchpad without having to shift their focus from the screen.
Mechanics	This topic describes the mechanics for the industrial design of Windows Precision Touchpads.

The guidance in this topic ensures that Windows Precision Touchpad interactions are natural and enhance the overall quality of the user experience. Windows Precision Touchpads are built for the user and are seamlessly integrated into a mobile computing experience.

Windows Precision Touchpads are optimized for Windows 8.1. The experience that precision touchpads provide is consistent and reliable, fast and responsive, and avoids issues of accidental activation. This advanced software experience is combined with a hardware certification program and superb industrial design to produce the ideal touchpad experience.

Windows Precision Touchpads are an evolution in touchpad hardware and software technology. This topic does not mandate specific methods or materials, but instead describes the desired end-user experience on precision touchpads. In many cases, this can be achieved by simple enhancements, such as using scratch and smudge resistant coatings. Utilizing advanced manufacturing techniques and materials reinforces the overall quality experience.

Windows Precision Touchpads are designed for mobile form factors, to be used together with touch and optimized for their unique strengths as an input modality. These devices are built and designed for the enjoyment and comfort of the user. Characteristics such as the location and size of the touchpad are designed to be intuitive and easy for everyone to use.

Windows Hardware Certification Kit (HCK) requirements

The following characteristics are verified as part of the Windows HCK for precision touchpads and have mandatory ranges:

- Placement
- Dimension
- Alignment
- Activation force

Glossary

Expand table

Term	Definition
Activation Force	Force exerted by a user that invokes a button report
Coversheet	Top layer of material overlaying the touchpad sensor
Edge Demarcation	Boundary marker for touchpad sensor active area

Term	Definition
Palm Deck	Region in which palms rest below the keyboard
Right Click Zone	Region that results in a right click when activation force is exerted upon it
Zone Marker	Identification marker delineating right click zone

Related topics

Design Recommendations for Windows 8 Touchpads

Surface Treatment (touchpad-surface-treatment)

Article • 01/24/2024

The coversheet of a Windows Precision Touchpad delights the user by its feel. The coversheet is manufactured with a high quality coating and machined texture to produce a glass-like smoothness.

Coating

A precision touchpad shall be protected with a surface coating that is scratch and smudge resistant. Scratch resistance ensures that the user experience is not hindered by undesired textural deformations and smudge resistance reinforces the polished aesthetic appeal. Finger prints and grease marks are undesirable and give the impression of an unclean, lower quality surface. Therefore, precision touchpads enhance the user experience by utilizing an oleophobic coating, ensuring the touchpad always looks pristine.

Texture

Regardless of surface coating or coversheet material, a precision touchpad shall have a smooth surface with no perceivable bumps or grooves.

Textured surfaces such as dimples or bumps increase the surface friction when the user glides over the touchpad. This added friction negatively impacts the user experience by slowing the user's motions, adding unnecessary finger fatigue from the additional force that is needed during interactions, and impedes precise cursor positioning and gesturing.

The coversheet shall have a perceived smoothness to glass. We recommend a surface roughness of < 0.01 Ra so that gestures feel responsive and user confidence is maximized.

Touchpad Palm Deck Integration (touchpad-palm-deck-integration)

Article • 01/24/2024

Integration of a precision touchpad with the palm deck impacts the ease of use of the touchpad. Precision touchpads are designed to be natural and intuitive interface for all users. Users should able to easily identify the interaction surface, gesture in a natural and responsive fashion, and seamlessly interact with the touchpad without having to shift their focus from the screen.

Placement

The optimal placement for a precision touchpad is centered between the *home keys*. For example, with an English keyboard the optimal touchpad placement is centered between the **F** and **J** keys (see *Figure 1 Optimal Horizontal Placement*). This is ideal because it is easy for users to naturally interact with the touchpad while typing, and without having to shift their focus from the screen to the touchpad.

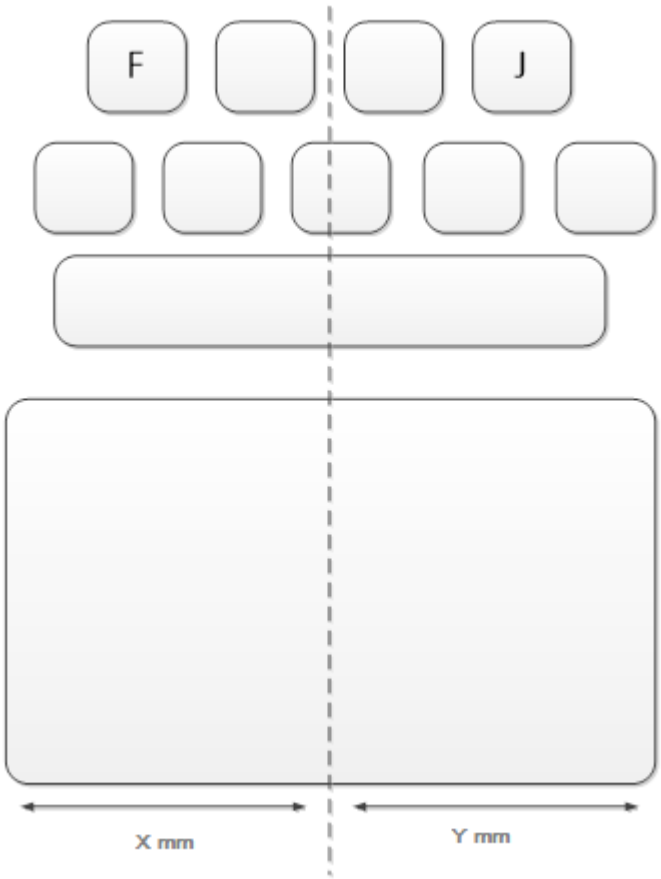


Figure 1 Optimal Horizontal Placement

Buttons

A precision touchpad will have no distinct physical buttons. Precision touchpads support the click action directly (by using a click-pad or pressure-pad). Additional buttons are unnecessary and redundant. Buttons take valuable space away from the gesture surface, add cost, and clutter the palm deck and remove focus from the touchpad itself. No touchpad buttons are permitted on the palm deck or keyboard alongside a precision touchpad.

Dimensions

The optimal and recommended size for the sensor region of a precision touchpad is defined as 65mm x 105mm (see *Figure 2 Precision Touchpad Optimal Size*). These dimensions provide the optimal user experience in which a user can comfortably perform touchpad gestures and control cursor position without repeatedly clutching (lifting the finger from the touchpad surface and then repositioning it). Smaller dimensions hinder the experience of gestures such as panning, zooming, and swiping from the edge, and force the user into clutching.

Figure 2 Precision Touchpad Optimal Size

The Windows Hardware Certification Kit (HCK) requirements mandate a minimum sensor size of 32mm x 64mm for precision touchpad devices, as shown in *Figure 3 Precision Touchpad Minimum Size*.

Figure 3 Precision Touchpad Minimum Size

Alignment

The precision touchpad is integrated so that it is level with the palm deck and there is no perceivable depth or delta. *Figure 4 Optimal Depth Offset* shows a side view of the touchpad as seen from the edge of the device. Due to manufacturing and integration tolerances, an allowance of 1.5mm is permitted for the depth offset. However, we recommend as close to 0mm depth offset (preferably 0-0.2mm as shown in *Figure 4 Optimal Depth Offset*) as possible, for the ideal user experience.

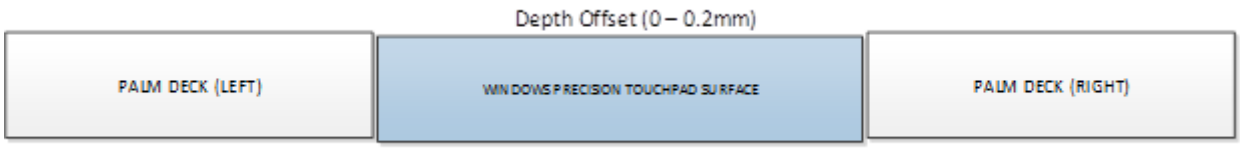


Figure 4 Optimal Depth Offset

Coversheet and palm deck gap

The region (gap) between a precision touchpad and the palm deck should be minimal, but perceivable by the user. A gap no greater than 0.3mm between the touchpad and the palm deck is acceptable (see *Figure 5 Gap Between Coversheet and Palm Deck*); this provides a comfortable user experience and allows the user to perform edge gestures without shifting focus from the screen.

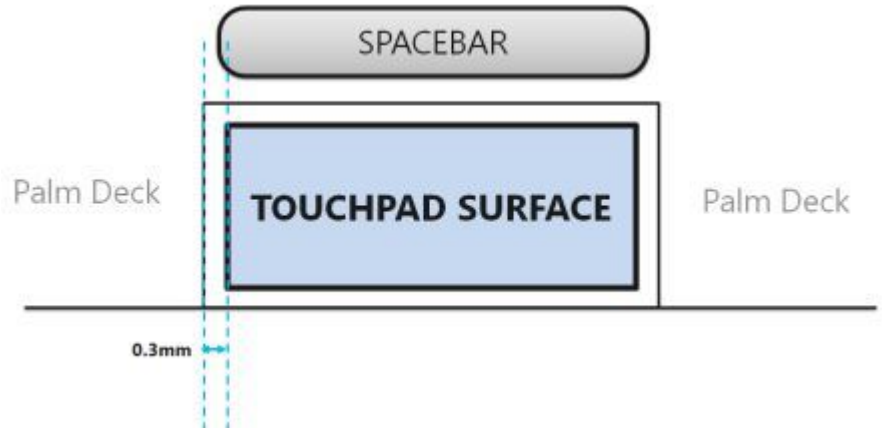


Figure 5 Gap Between Coversheet and Palm Deck

Edge demarcation

For implementations where the user cannot tactile-perceive the edge of the active sensor area, an edge demarcation can be used on the coversheet. Edge demarcations can be located along the boundary of the touchpad directly on the coversheet, and shall not start more than 1mm from the coversheet edge as shown in *Figure 6 Embossing Region*. Manufacturers can choose any suitable technique to make the edge demarcation perceivable to the user, such as texturing or embossing.

Figure 6 Embossing Region

Zone markers

Precision touchpads implement right-click zones. To clearly identify this region to users, zone identification markers are permitted. If implemented, zone markers shall be located in the lower center region of the precision touchpad. The zone marker shall be 10mm in height or 25% of the vertical dimension of the precision touchpad sensor region (whichever is larger, see *Figure 7 - Zone Marker for Small Precision Touchpad* and *Figure*

8 Zone Marker for Large Precision Touchpad). This delineation clearly identifies the right click region on the precision touchpad.

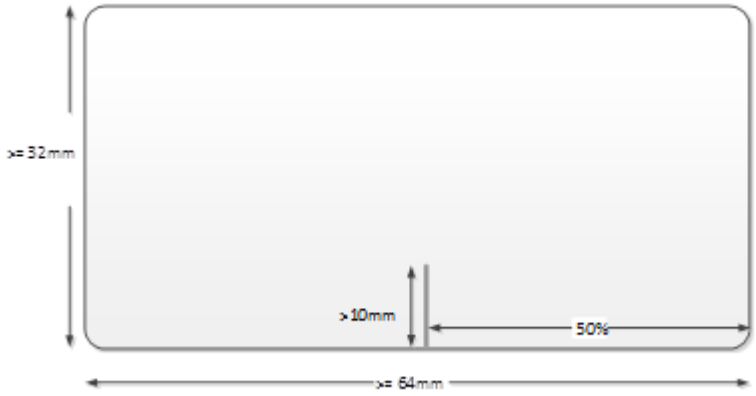


Figure 7 - Zone Marker for Small Precision Touchpad

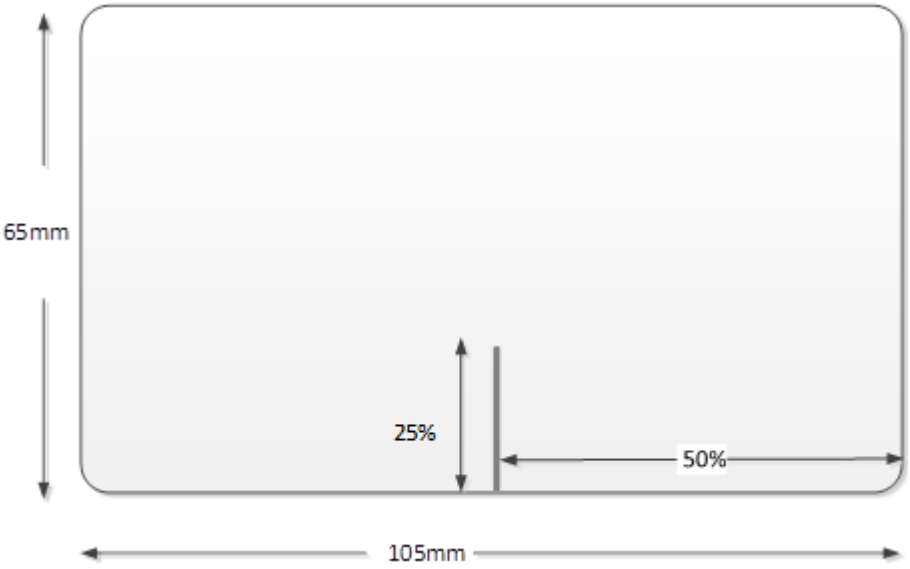


Figure 8 Zone Marker for Large Precision Touchpad

Mechanics (touchpad-mechanics)

Article • 01/24/2024

This topic describes the mechanics for the industrial design of Windows Precision Touchpads.

Activation force

Activation force is defined as the required force to invoke a button report from a touchpad device. To ensure a consistent experience across all precision touchpad devices, minimum activation forces are defined and validated in the WHCK.

Click-pad implementations

Precision touchpads that are implemented as a click-pad shall strive to provide uniform activation across the touchpad surface such that 150g will result in a button report anywhere. However, due to the varying designs and costs of associated hinge mechanisms, the activation force gradient that is shown in *Figure 1 Click-pad Activation Force* is a good starting point.

The Windows Hardware Certification Kit (HCK) requires and validates activation force specifically in the lowest 10mm / 25% region (whichever is largest) of the touchpad.

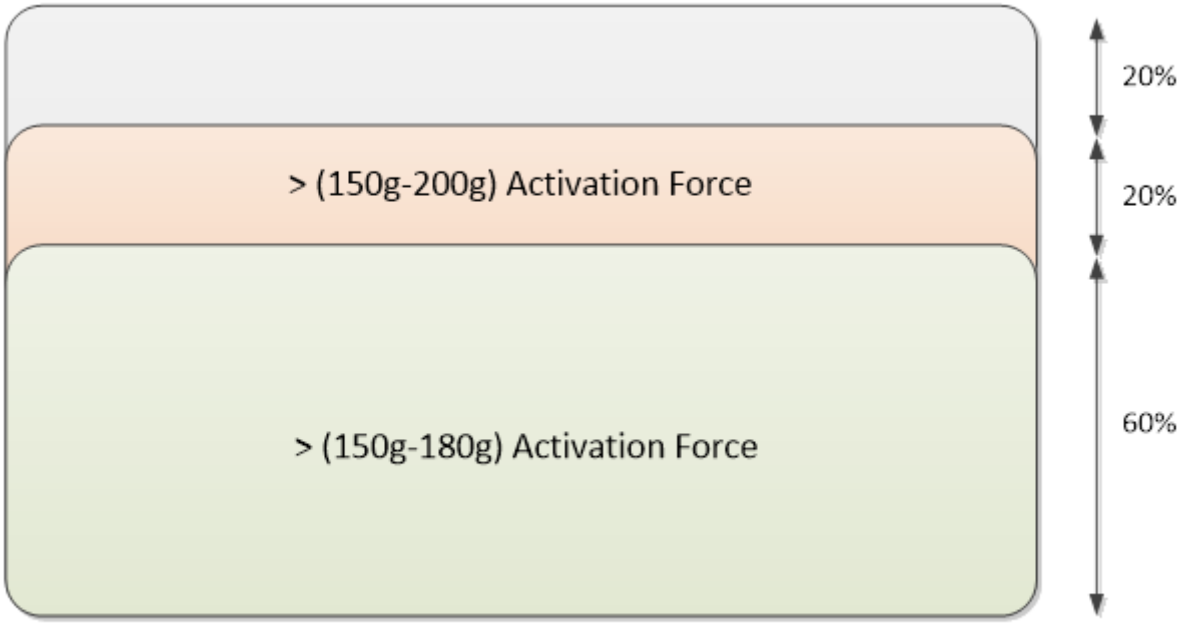


Figure 1 Click-pad Activation Force

Pressure-pad

Precision touchpads that are implemented as a pressure-pad shall strive to provide uniform activation across the touchpad surface, such that 150g will result in a button report anywhere. There is a greater expectation of uniform activation on these devices due to the lack of hinges and lack of tactile feedback to the user. It is prudent to provide haptic or audible feedback to the user when activation force has been exerted.

The Windows HCK requires and validates activation force specifically in the lowest 10mm / 25% region (whichever is largest) of the touchpad.

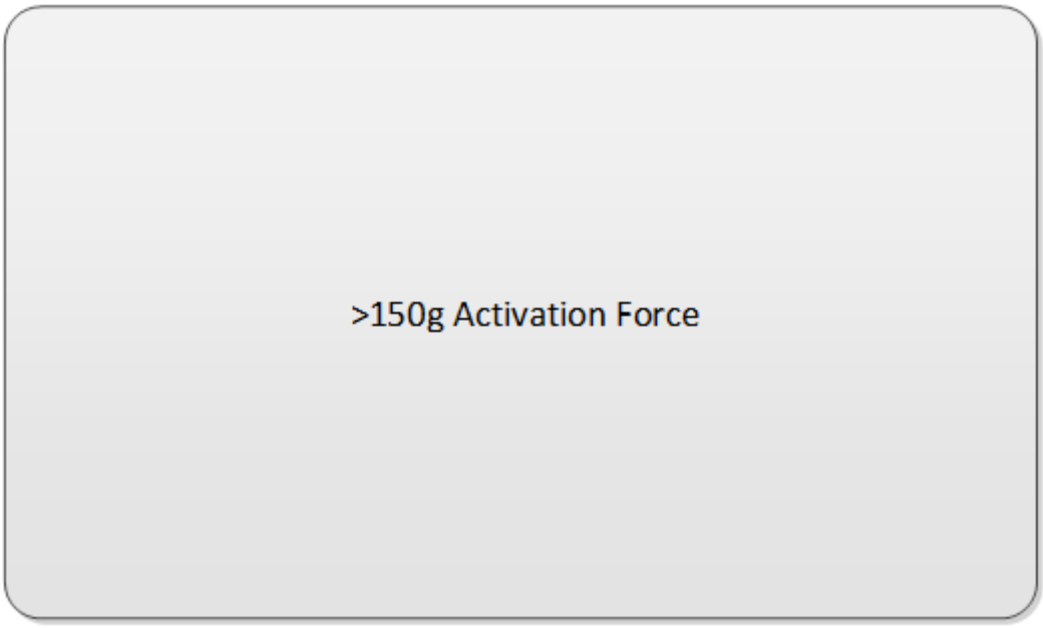


Figure 2 Pressure-pad Activation Force

Surface Deflection

The amount in which a surface flexes or bends when under force is defined as surface deflection. Surface deflection can vary based on the methods and materials that are used in manufacturing a touchpad coversheet. The acceptable range for precision touchpad surface deflection is less than 0.05mm. The coversheet should not bend more than 0.05mm anywhere on the device when the user is exerting normal forces that occur during button activation and interaction, as shown in *Figure 3 Surface Deflection* and *Figure 4 Surface Corner Deflection*. High degrees of deflection feel unnatural to the user and lead to perceived fragility and quality issues.

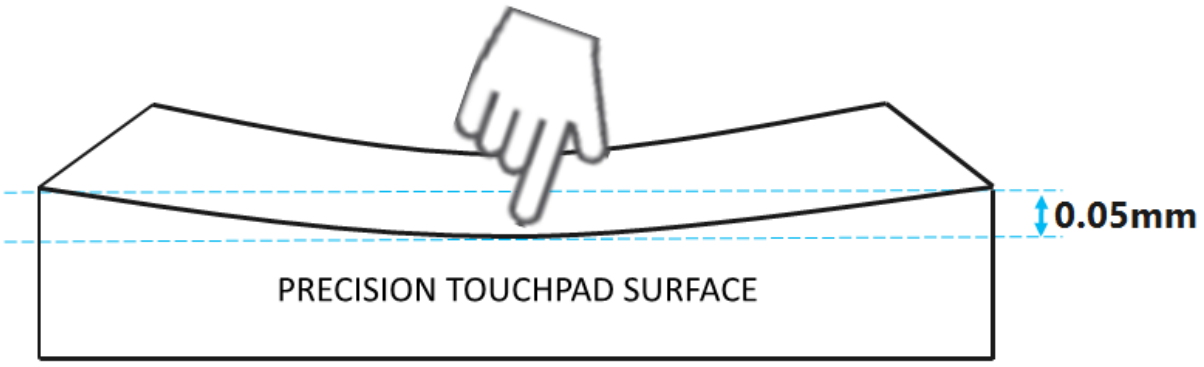


Figure 3 Surface Deflection

Figure 4 Surface Corner Deflection

Click depth (click-pad devices)

Applying activation force to a precision touchpad click-pad device causes the coversheet to depress. The total depression, including deflection of the coversheet, shall be no greater than 0.3mm. When viewed from an angle looking down on the device, *Figure 5 Click Depth* shows that the click depression does not exceed 0.3mm.

This depth provides an optimal click travel distance and has a natural feel to the user. This click depth also prevents foreign objects,including fingers, from becoming trapped in the gap between the touchpad and palm deck.

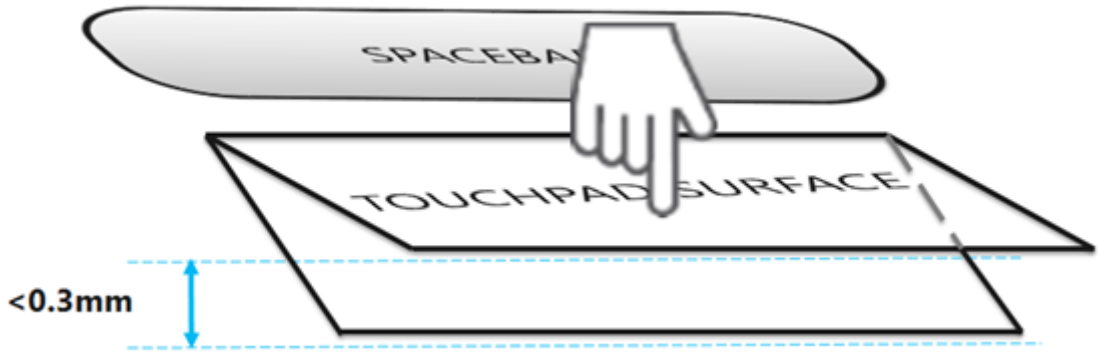


Figure 5 Click Depth

Legacy Touchpad PC Settings Opt-In (touchpad-legacy-touchpad-pc-settings-opt-in)

Article • 01/24/2024

This topic describes how devices that are not Windows Precision Touchpads can opt in to various settings that are exposed in Windows 8.1 to provide a simple and easy-to-navigate inbox solution to manage the most common touchpad settings.

PC Settings in Windows 8.1 exposes the following four settings for Windows Precision Touchpads:

- Enable/disable
- Edge gesture enable/disable
- Scroll direction normal/inverted
- Strength of inbox accidental activation prevention (AAP)

Legacy touchpad helper applications can opt-in to the enable/disable setting, and optionally the edgy enable/disable and scroll direction settings.

Opt-in requirements

1. To participate in the opt-in program, a touchpad applet must be able to toggle the enable/disable touchpad setting.
2. The opt-in program requires setting a registry value and consuming registry changes at run-time to change touchpad behavior. This mechanism is only available to user-mode helper applications that are associated with legacy touchpads, and is therefore not supported on WoA (Windows on Arm) platforms.
3. This opt-in program is specifically targeted to clamshell and convertible systems and is not supported on slate systems that provide an external dock with a legacy touchpad, because those systems can switch accessories that can have differing capability levels.

Opt-in mechanism

To specify which settings a touchpad applet can consume through the opt-in program, you must specify a DWORD under the following registry key, with the name string equal to the HWID string for the device for which it is opting in. This opt-in is global; that is, all users will see the opted-in settings as available in the modern settings applet.

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\LegacyControlled

The value of that DWORD specifies which settings are to be consumed by the applet in the form of a bitmask.

Expand table

Setting	Value	Mandatory/Optional
Enable/Disable	0x00000001	Mandatory
Edgy Enable/Disable	0x00000002	Optional
Scroll Direction	0x00000004	Optional

For example, for a device that has HWID (ACPI\MSFT0001) to opt in to just enable/disable and scroll direction, it would create a DWORD as follows:

ACPI\MSFT0001 0x00000005

If a modern touchpad with HWID (HID\VID_045E&PID_003F&REV_03FF) wanted to opt in to all settings, it would create a DWORD as follows:

HID\VID_045E&PID_003F&REV_03FF 0x00000007

Note If you import or export from the registry, the "\" will be duplicated as "\\" because this is a delimiting character. If you paste by using the Registry Editor, ignore the extra delimiter.

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouch
Pad\LegacyControlled]
"HID\\VID_045E&PID_003F&REV_03FF"=dword:00000007
"ACPI\\MSFT0001"=dword:00000005
```

For more information, see HIDClass Hardware IDs for Top-Level Collections.

Settings change consumption

Although the opt-in by the touchpad helper application is global, the settings themselves are customized on a per-user basis. To consume the settings changes, the

helper application must register for registry change notifications that are associated with the following key and its sub-keys and act on those changes.

If there is no logged-on user, default settings should be used so that the touchpad is enabled and scroll direction is normal (not inverted).

HKCU\Software\Microsoft\Windows\CurrentVersion\PrecisionTouchPad

To do this, use the RegNotifyChangeKey() function and specify the bWatchSubtree parameter as TRUE.

Whenever a touchpad setting is modified by the user by using the inbox PC settings, the event handle that the call specifies to RegNotifyChangeKey() is used to notify the caller.

When that event occurs, the touchpad applet shall read each of the following values as appropriate per opt-in value specified.

HKCU\Software\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\Status

Expand table

Setting	Name	Type	Values
Enable/Disable	Enabled	DWORD	0 – Disabled Non-0 – Enabled

HKCU\Software\Microsoft\Windows\CurrentVersion\PrecisionTouchPad

Expand table

Setting	Name	Type	Values
Edgy Enable/Disable	EnableEdgy	DWORD	0 – Disabled Non-0 – Enabled
Scroll Direction	ScrollDirection	DWORD	0 – Standard Non-0 – Reverse

Note The values that are named in the preceding tables should only be read and never overwritten by the opt-in entity; see the following section about third-party touchpad control panels.

Third-party touchpad control panels

Any touchpad setting that has been opted in by the helper application can be displayed by a third-party control panel. However, to ensure consistency and synchronization with the inbox settings user interface (UI), the display should not be modifiable. Any custom hotkeys that were used to change settings (for example, enable/disable) should be non-functional if they impact any opted-in setting. Windows provides an inbox hotkey for enable/disable control (**CTRL+WIN+F24**), which an OEM can support .

Uninstall

If the touchpad helper application that handles the settings changes is uninstalled, the uninstall procedure should remove the opt-in that was specified under **HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\LegacyControlled**.

Legacy Touchpad Forced Detection (touchpad-legacy-touchpad-forced-detection)

Article • 01/24/2024

This topic describes the mechanism for forcing Windows 8.1 to detect a touchpad irrespective of connectivity as a legacy device on mobile form-factors.

Any touchpad that is not identified as a certified Windows Precision Touchpad is classified as a legacy touchpad provided it is identified by the host as an integrated peripheral on a mobile form-factor. If an integrated touchpad is not identified by the host as a legacy touchpad, the following features will not be available:

- Inbox Accidental Activation Prevention (AAP)
- PC Settings Touchpad Page
  - Including the ability to opt-in to settings beyond AAP
  - See Legacy Touchpad PC Settings Opt-In for additional information

There are circumstances in which a legacy touchpad integrated with a keyboard may not be identified correctly due to underlying issues in how the device is exposed to the host.

1. Touchpad is connected via a USB port marked as removable
2. Touchpad is connected via Bluetooth

In the above circumstances, Windows 8.1 will automatically assume the touchpad is an external mouse or touchpad and will not enable legacy touchpad features.

Approved List Mechanism

Irrespective of how a device is exposed to the host, Windows 8.1 provides the ability to force legacy identification and feature enablement through an approved list mechanism.

By creating a registry entry under the following key and specifying the desired matching level, Windows 8.1 will force a matching device to be detected as a legacy touchpad.

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchPad\LegacyDevices

The value of that DWORD specifies what level of matching should be undertaken for a specific entry.

Expand table

Matching Level	Value
VID/PID Only Match	0x00000001
VID/PID/REV Match	0x00000002
Full HWID Match	0x00000003

For example if a device with HWID (ACPI\MSFT0001) wanted to be exactly matched based on HWID, it would specify its entry as follows:

ACPI\MSFT0001 0x00000003

For example if a touchpad with HWID (HID\VID_045E&PID_003F&REV_03FF&MI_02&COL01) wanted to ensure a match irrespective MI and COL, it would specify its entry as follows:

HID\VID_045E&PID_003F&REV_03FF 0x00000002

For more information, see HIDClass Hardware IDs for Top-Level Collections.

With reference to the above example, if the same touchpad wanted to ensure a match irrespective of version (RevID), it would specify its entry as follows:

HID\VID_045E&PID_003F 0x00000001

Note that a full HWID string may be specified in the registry with the DWORD value indicating the level of matching, for instance the second example has a functional equivalent below:

HID\VID_045E&PID_003F&REV_03FF&MI_02&COL01 0x00000002

Note If importing or exporting from the registry, the "\" will be duplicated as "\\" since this is a delimiting character. If pasting via registry editor, ignore the extra delimiter.

```
Windows Registry Editor Version
5.00``[HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\PrecisionTouchP
ad\LegacyDevices] "HID\\VID_045E&PID_003F&REV_03FF"=dword:00000002
```

Precision Touchpad Validation Guide (touchpad-validation-guide)

Article • 01/24/2024

This section is a validation guide for a Windows Precision Touchpad device, for Windows 10 and later operating systems.

The guidance provided here ensures compliance with Hardware Lab Kit (HLK) testing for a Windows Precision Touchpad device.

In this section

Expand table

Topic	Description
General Testing Guidelines	This topic provides general guidelines for testing a Windows Precision Touchpad device, using the Windows Hardware Lab Kit (HLK) for Windows 10.
Precision Touchpad Tests	This section lists the topics that provide details about the Precision Touchpad tests in the Windows Hardware Lab Kit (HLK) for Windows 10.
Hardware Requirements and Vendor Information	This topic provides a list of the test equipment that is required for the touch-related tests in the Windows Hardware Lab Kit (HLK), and information about the vendors who stock the listed equipment.
Appendix	The appendix for the Windows Precision Touchpad Validation Guide, provides information about common error messages, along with some HID-specific ones.

General Testing Guidelines (touchpad-general-testing-guidelines)

Article • 01/24/2024

This topic provides general guidelines for testing a Windows Precision Touchpad device, using the Windows Hardware Lab Kit (HLK) for Windows 10.

Testing Guidelines

- Unless otherwise specified, the device should always have AC power when performing HLK tests.
- Unless otherwise specified, all tests should be run with "Test Signing Mode" set to ON. To do this, perform the following steps:

    1. Open a **Command Prompt** window as an Administrator.
    2. Enter this command: **bcdedit -set testsigning on**
    3. Restart the computer.

- Unless otherwise specified, use 7mm diameter contacts for tests requiring use of the PT3 (Precision Touch Testing Tool).
- Unless otherwise specified, all numbers in error messages which report distance or location are in himetric units (0.01mm).
- The grids in the PTLogo visualization help align vertices (for tests like Linearity) and graph distances. The distance from one line to the next 200 himetric units (in other words, is 2mm).
- Unless otherwise specified, multiple contacts should maintain a minimum separation distance: For all alignments, contacts should be at least 8mm apart, edge to edge during test.
- Regarding the GetThqaBlob test, note the following:

    o Path to *GetThqaBlob.exe*: C:\Program Files (x86)\Windows Kits\10\Hardware Lab Kit\Tests\x86\digitizer\Win8Touch.

    o File location management: All necessary files are copied to test machines when the HLK client is installed.

- Due to the nature of customization drivers, it is likely that they will cause test failures for devices during compatibility tests. The HLK tests are intended to test hardware capabilities, and the filter driver may interrupt the expected data flow to the tests.

  If you want to ship your devices with customization, then you should include the customization components during testing. But you must make sure that these custom behaviors can be disabled during testing, if they would otherwise block the device from passing certification.

PTLogo Interface

- To manually pass an iteration (where applicable), press “P” on the keyboard.
- To manually fail an iteration (where applicable), press “F” on the keyboard.
- To manually restart the test (where applicable), press “R” on the keyboard.
- To manually fail the test (where applicable), press “T” on the keyboard.
- To manually advance to the next iteration (where applicable), press “N” on the keyboard.
- To exit PTLogo, press “E” on the keyboard.

Precision Touchpad Tests (touchpad-tests)

Article • 01/24/2024

This section lists the topics that provide details about the Precision Touchpad tests in the Windows Hardware Lab Kit (HLK) for Windows 10.

In this section

Expand table

Topic	Description
Aliasing	This is to test the contact reporting capabilities of a Windows Precision Touchpad device, with respect to aliasing.
All Areas Reported	This is to test the reporting area coverage of a Windows Precision Touchpad device.
Buffering	This is to test the data buffering behavior of a Windows Precision Touchpad device.
Confidence Reporting	This is to test the confidence reporting capabilities of the Windows Precision Touchpad device.
Converge / Diverge	This is to test the contact tracking and reporting capabilities of a Windows Precision Touchpad device.
Device Button Reporting	This is to test the integrated buttons in a Windows Precision Touchpad device.
Device Height	This is to validate the height of a Windows Precision Touchpad device.
Device Type	This is to test the pad type reporting of a Windows Precision Touchpad device.
Device Width	This is to validate the width of a Windows Precision Touchpad device.
External Buttons	This is to test that the buttons attached to a Windows Precision Touchpad device, are reported correctly.

Topic	Description
Ghost Reporting	This is to test the accuracy of the reported contacts for a Windows Precision Touchpad device.
Input Resolution	This is to validate that the Windows Precision Touchpad device is able to report at its specified resolution.
Latency - Panning	This is to test the reporting capabilities of the Windows Precision Touchpad device, with regard to latency and panning.
Latency - Touch Down	This is to test the reporting capabilities of the Windows Precision Touchpad device, with regard to latency and touch down.
Linearity	This is to test the linearity performance of a Windows Precision Touchpad device.
Maximum Supported Contacts	This is to test the maximum number of contacts supported by a Windows Precision Touchpad device.
Positional Accuracy	This is to test the positional accuracy for reporting contacts on a Windows Precision Touchpad device.
Positional Accuracy - Manual	This is to test the positional reporting accuracy of a Windows Precision Touchpad device.
Power State Transitions	This is to test the behavior of a Windows Precision Touchpad device during power state transitions.
PTPHQA	This is to test the PTPHQA reporting capabilities of a Windows Precision Touchpad device.
Report Rate - Single Contact	This is to test the single contact reporting capabilities of a Windows Precision Touchpad device.
Report Rate - Multiple Contacts	This is to test the multiple contact reporting capabilities of a Windows Precision Touchpad device.
Right Click Reliability	This is to test the right-click detection capabilities of a Windows Precision Touchpad device.

Topic	Description
Selective Reporting	This is to test the selective reporting capabilities of a Windows Precision Touchpad device.
Static Validation	This is for performing static validation tests.
Stationary Jitter	This is to test the stationary contact reporting abilities of a Windows Precision Touchpad device.
Z-Height Detection	This is to test the z-axis reporting behavior of a Windows Precision Touchpad device.

Aliasing (touchpad-aliasing)

Article • 01/24/2024

This is to test the contact reporting capabilities of a Windows Precision Touchpad device, with respect to aliasing.

Test name

- Test.Aliasing.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.FingerSeparation

Test purpose

- Verifies that the device can track and report unique, continually arriving and departing contacts without aliasing at fixed separation distances.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.Aliasing.json.
2. Place two fingers approximately10mm apart, horizontally.
3. Alternating between the two fingers, tap the touchpad at 240 taps/minute. When tapping, hold down a contact until it is time to make another tap with that contact (in order to interleave the down times of the two contacts), rather than making light taps, one after another.
4. Verify that after tapping for 10 seconds, there is no line drawn between these contacts (see Common Errors for an example of a failure).
5. Repeat **Step 2** to **Step 5** with fingers aligned on the vertical axis.
6. Repeat **Step 2** to **Step 5** with fingers aligned on the diagonal axis.
7. Manually pass the iteration, if validation in every axis was successful.
8. Repeat **Step 2** to **Step 7** with fingers approximately 33mm apart.

Note If the Precision Touchpad device has a height of less than 33mm, you should verify vertical aliasing at the maximum possible distance.

Common error messages

- "[20] Positional delta too large" o Aliasing caused device contact swap, thus causing device to report a quick swipe. o The value represents the displacement distance. o Any displacement greater than 270 himetric units (2.7mm) will result in this error.

Here's a screenshot from the Aliasing test, showing a displacement distance that is too large.

Passing criteria

- 2/2 (100%) iterations must pass in order to complete with passing status.

All Areas Reported (touchpad-all-areas-reported)

Article • 01/24/2024

This is to test the reporting area coverage of a Windows Precision Touchpad device.

Test name

- Test.AllAreas.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Accuracy

Test purpose

- Verifies that the device can report contacts in all areas of the digitizer surface.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.AllAreas.json.
2. Following the directions on the screen, you will perform 5 taps in each general region of the touchpad (top-left, top-right, bottom-left, bottom-right, and center):
   a. Tap anywhere inside the specified region. For example, if top-left is specified, tap anywhere left of center and above the middle of the touchpad.

i. Tap a different random location for each of the 5 iterations.

ii. If the tap occurs in the expected region, PTLogo will automatically pass the iteration.

iii. If no tap is registered for a given region, manually fail the iteration.

b. After 5 taps, the onscreen instructions will indicate the next region to tap. Repeat the preceding **Step 2.a** for each subsequent region as indicated. **Common error messages**

- None.

Passing criteria

- 25/25 (100%) iterations must pass in order to complete with passing status.

Buffering (touchpad-data-buffering)

Article • 01/24/2024

This is to test the data buffering behavior of a Windows Precision Touchpad device.

Test name

- Test.Buffering.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Buffering

Test purpose

- Verifies that the device has no data loss on wake from low power states.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.Buffering.json.
2. Follow the directions on the screen and perform a swipe with the maximum number of supported fingers on the touchpad.
3. The swipe should start at the top of the touchpad, and continue past the center of the device – this motion should be quick, no longer than a half second.
4. The test will pass if the swipe was fully registered.
5. Repeat this test 5 times.

Common error messages

- None.

Passing criteria

- 5/5 (100%) iterations must pass in order to complete with passing status.

Confidence Reporting (touchpad-confidence-reporting)

Article • 11/14/2024

This is to test the confidence reporting capabilities of the Windows Precision Touchpad device.

Test name

- Test.Confidence.json

Core requirements tested

- Device.Input.Digitizer.Base.HIDCompliant

Test purpose

- Verifies that the device can report a lack of confidence (by turning off the confidence bit for a contact), when a contact exceeds the sizing of an intentional finger contact.

Tools required

- PTLogo.exe
- A 140g or 190g slug ordered from ITRI or Triex. See Hardware Requirements and Vendor Information for vendor information.

Validation steps

1. Launch Test.Confidence.json.
2. Place a 27mm-29mm diameter contact straight down onto the surface of the touchpad for at least 5 seconds.
3. Remove the contact. PTLogo will automatically verify that the confidence bit was cleared in the required time, and remained cleared for the contact lifetime. If no reporting errors were detected, it will automatically pass the iteration.

Common error messages

- "[30] confidence bit set too long." The device is not clearing the confidence bit within 50ms of contact reporting.
- "[7] test didn’t receive enough data." User lifted off of the touchpad too early.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Converge / Diverge (touchpad-converge---diverge)

Article • 01/24/2024

This is to test the contact tracking and reporting capabilities of a Windows Precision Touchpad device.

Test name

- Test.ConvergeDivergeDiagonal.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.FingerSeparation

Test purpose

- Verifies that the device can track and report unique contacts without aliasing at continually varying separation distances.

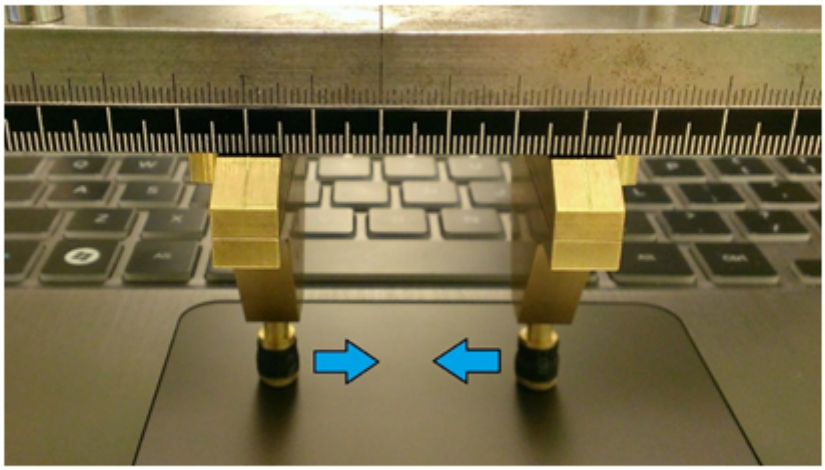
Tools required

- PT3 w/Precision Touchpad Modifications (or similar assembly for converge/diverge).
- PTLogo.exe

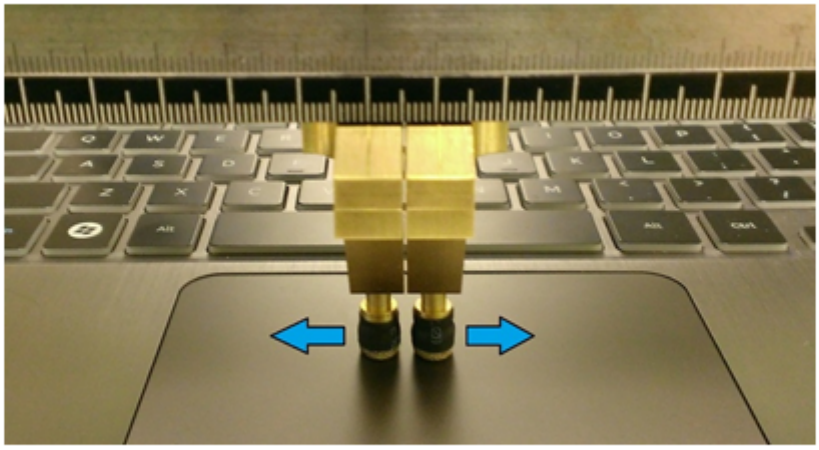
Validation steps

1. Launch Test.ConvergeDivergeDiagonal.json in ptlogo.exe.
2. Use the converge/diverge assembly with two 7mm precision contacts.
3. Rotate the plate 45°.
4. Place these precision contacts on the digitizer surface separated by distance as instructed on screen.
5. As instructed, move contacts either towards each other (converge) or apart (diverge). a. Every 2nd iteration, move the contacts to a new location on the touchpad. b. Five iterations per direction for both converge and diverge (10 total).
6. If the 2 contacts remain distinct and no errors are detected, PTLogo will automatically pass the iteration and advance to the next.

- As shown in the following image, start with contacts greater than 23mm apart, and move them together until they are 8mm apart, edge to edge.



- As shown in the following image, start with contacts 9mm apart edge to edge, and move them until they are greater than 23mm apart.



Common error messages

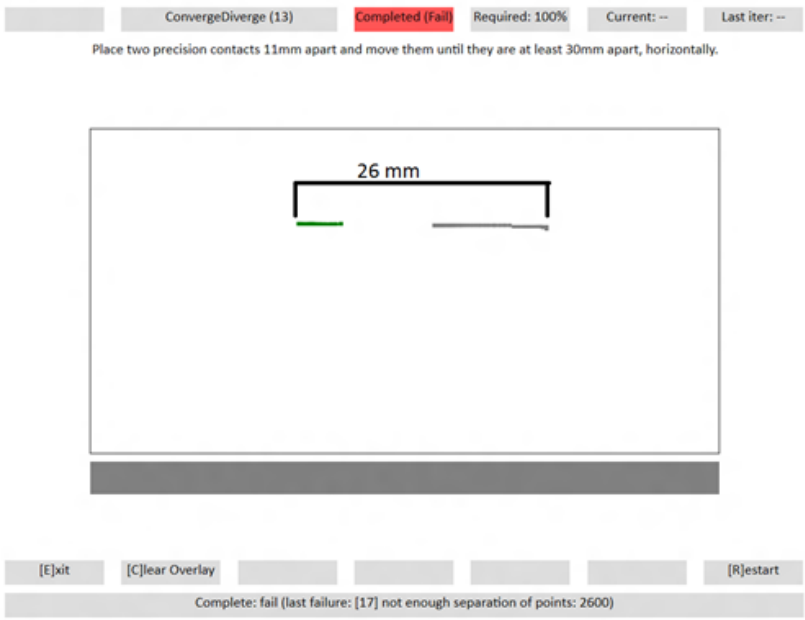
When you perform this test, make sure that the widest distance between contacts is always greater than 23mm, and the smallest distance is always less than 9mm from edge to edge.

Here are some common error messages, along with their meanings.

- "[17] not enough separation of points: ####."

Either a converge test didn’t start with the contacts far enough apart, or a diverge test didn’t pull the contacts apart enough. The displayed value (####) indicates the reported distance between the contacts.

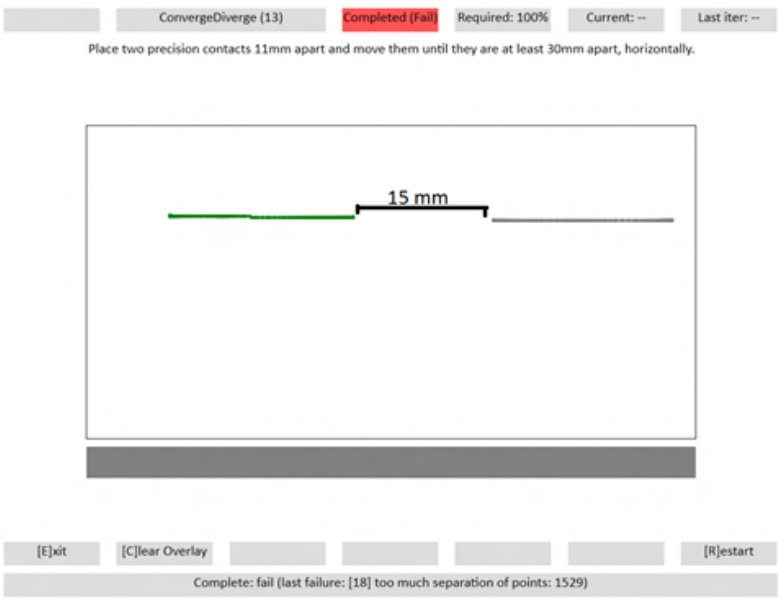
Here's a screenshot from the test, showing insufficient separation between points.


ConvergeDiverge (13)
Completed (Fail)
Required: 100%
Current: --
Last iter: --
Place two precision contacts 11mm apart and move them until they are at least 30mm apart, horizontally.
26 mm
[E]xit
[C]lear Overlay
[R]estart
Complete: fail (last failure: [17] not enough separation of points: 2600)

- "[18] too much separation of points: ####."

  Either a converge test didn’t bring the contacts close enough together, or a diverge test didn’t start with the contacts close enough. The displayed value (####) indicates the reported distance between the contacts.

  Here's a screenshot from the test, showing too much separation between contact points.


ConvergeDiverge (13)
Completed (Fail)
Required: 100%
Current: --
Last iter: --
Place two precision contacts 11mm apart and move them until they are at least 30mm apart, horizontally.
15 mm
[E]xit
[C]lear Overlay
[R]estart
Complete: fail (last failure: [18] too much separation of points: 1529)

Passing criteria

- A total of 30/30 (100%) iterations must pass in order to complete with passing status.

Device Button Reporting (touchpad-device-button-reporting)

Article • 01/24/2024

This is to test the integrated buttons in a Windows Precision Touchpad device.

Test name

- Test.DeviceButton.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Buttons

Test purpose

- Verifies that the device reports button presses correctly for integrated buttons.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.DeviceButton.json.
2. *Iteration One*: Button presses with capacitive contacts:

a. Place a finger in the center area of the digitizer surface, and exert sufficient activation force to depress the physical button. Verify that the button state is down (the grey bar at the bottom of PTLogo tool should turn green), and that the button state returns to off when your finger is lifted from the button.

b. Repeat the preceding step with 2, 3… N contacts, where N is the maximum number of contacts supported by the device. Contacts should always maintain a minimum separation distance of 15mm.

c. Place 2 fingers on the touchpad surface, and press down on the physical button. Keeping the button pressed with one finger, lift the second. PTLogo with automatically check that the button state is reported consistently.

d. Depress the physical button with three fingers on the touchpad surface. Maintaining downward pressure, flick the fingers downward so that they move off the touchpad.

Verify that the physical button state does not stay on, and that the bar beneath the PTP visualization returns from green to grey once the fingers are off the touchpad.

e. Upon successful validation, manually pass the iteration. Otherwise, manually fail the iteration.

Common error messages

- None.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status

Device Height (touchpad-device-height)

Article • 01/24/2024

This is to validate the height of a Windows Precision Touchpad device.

Test name

- Test.DeviceHeight.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Dimensions

Test purpose

- Verifies that the device sensor meets or exceeds minimum height requirement of 32mm.

Tools required

- Ruler or Micrometer
- PTLogo.exe

Validation steps

1. Launch Test.DeviceHeight.json.
2. Measure touchpad height with a ruler or micrometer.
3. Assuming Y is the reported sensor width, verify that:
    a. The measured device width is between Y and (Y + 4).
    b. Y is greater than or equal to 32mm.
4. If these conditions are true, then manually pass the iteration, otherwise manually fail the iteration.

Because the sensor may be slightly smaller than the cover sheet, the measured height may be as much as 4mm greater than the height that is reported by the device.

> **ⓘ Important**
>
> To provide the best possible experience for users while mousing and gesturing, it is important to clearly indicate the start and end of the touchpad active area.

While we recommend that touchpads have a permanent physical indication of the active area to aide the original purchaser and future users of a device, at a minimum some temporary indication is required as part of retail unboxing to assist. If during an audit, said operator is unable to accurately measure the touchpad active dimensions, they will be unable to pass 1 or more HLK tests which may result in audit failure and revocation of HLK certification. Users need to have some bearings on where the active area of a touchpad starts and ends so they can be aware of the interaction surface for mousing and gesturing.

This guidance applies to all touchpad form factors, including but not limited to; seamless, extra buttons, dead zones, etc.

Common error messages

- None.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Device Type (touchpad-device-type)

Article • 01/24/2024

This is to test the pad type reporting of a Windows Precision Touchpad device.

Test name

- Test.DeviceType.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.DeviceTypeReporting

Test purpose

- Verifies that the device reports its pad type correctly.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.DeviceType.json.
2. If the device is a click-pad and PTLogo reports that the device is a click-pad, then manually pass the iteration.
3. If the device is a pressure-pad and PTLogo reports the device as a pressure-pad, then manually pass the iteration.
4. If the device is a non-clickable touchpad, or discrete-pad, and PTLogo reports the device as either Discrete Pad or No Pad Type, then manually pass the iteration.
5. If the device type reported in PTLogo does not match the actual device type, manually fail the iteration.

Common error messages

- None.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Device Width (touchpad-device-width)

Article • 01/24/2024

This is to validate the width of a Windows Precision Touchpad device.

Test name

- Test.DeviceWidth.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Dimensions

Test purpose

- Verifies that the device sensor meets or exceeds minimum width requirement of 64mm.

Tools required

- Ruler or Micrometer
- PTLogo.exe

Validation steps

1. Launch Test.DeviceWidth.json
2. Measure touchpad width with a ruler or micrometer.
3. Assuming X is the reported sensor width, verify that:
   a. The measured device width is between X and (X + 4).
   b. X is >= 64mm.
4. If these conditions are true, then manually pass the iteration, otherwise manually fail the iteration. Because the sensor may be slightly smaller than the cover sheet, the measured width may be as much as 4mm greater than that reported by the device.

> **ⓘ Important**
>
> To provide the best possible experience for users while mousing and gesturing, it is important to clearly indicate the start and end of the touchpad active area.
>
> While we recommend that touchpads have a permanent physical indication of the active area to aide the original purchaser and future users of a device, at a

minimum some temporary indication is required as part of retail unboxing to assist. If during an audit, said operator is unable to accurately measure the touchpad active dimensions, they will be unable to pass 1 or more HLK tests which may result in audit failure and revocation of HLK certification. Users need to have some bearings on where the active area of a touchpad starts and ends so they can be aware of the interaction surface for mousing and gesturing.

This guidance applies to all touchpad form factors, including but not limited to; seamless, extra buttons, dead zones, etc.

Common error messages

- None.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

External Buttons (touchpad-external-buttons)

Article • 01/24/2024

This is to test that the buttons attached to a Windows Precision Touchpad device, are reported correctly.

Test name

- Test.ExternalButtons.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Button

Test purpose

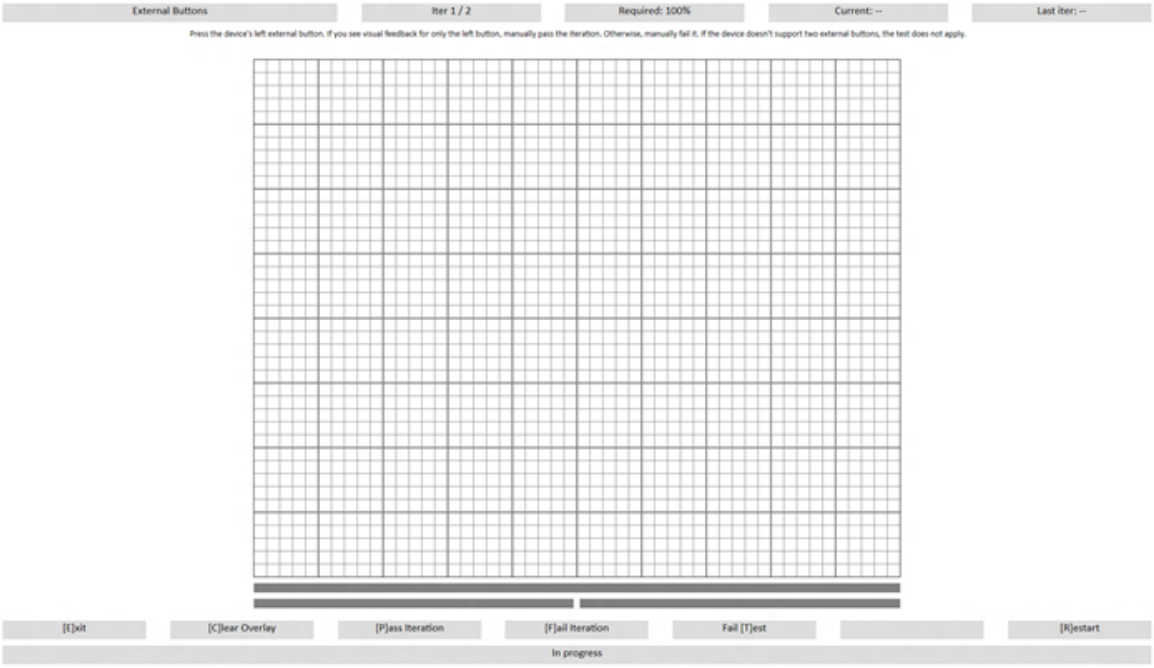
- Verifies that the attached buttons report correctly, to provide primary and secondary click functionality for a user.

Tools required

- Ruler or Micrometer
- PTLogo.exe

Validation steps

1. Launch Test.ExternalButtons.json.
2. If the device does not support external buttons, pass the test manually.
3. If the device is not a click pad or a pressure pad, then visually ensure that there are two buttons available.
4. Click each button and verify that the external button images in the UI, light up as each button is pressed. Note the two extra button bars as shown in the following screenshot. These will change color when the proper button messages are received.


External Buttons
Iter 1 / 2
Required: 100%
Current: --
Last iter: --
Press the device's left external button. If you see visual feedback for only the left button, manually pass the iteration. Otherwise, manually fail it. If the device doesn't support two external buttons, the test does not apply.
[E]xit
[C]lear Overlay
[P]ass Iteration
[F]ail Iteration
Fail [T]est
[R]estart
In progress

5. If both buttons generate the correct response, the test will pass. **Common error messages**

- None.

Passing criteria

- 1/1 (100%) iterations with each button must pass in order to complete with passing status.

Ghost Reporting (touchpad-ghost-reporting)

Article • 01/24/2024

This is to test the accuracy of the reported contacts for a Windows Precision Touchpad device.

Test name

- Test.GhostReporting.json

Core requirements tested

- Device.Input.Digitizer.Base.ContactReports

Test purpose

- Verifies that the device does not report false contacts (noise) when it is running on AC, or DC power (if applicable).

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.GhostReporting.json.
2. Make sure the system connected to the device is plugged in during this iteration.
3. Allow the test to run for 60 seconds, during which time you do not make any contact with the touchpad.

Note There is a timer in the top-right corner of the screen for reference.

4. PTLogo will automatically verify that no contacts have been reported by the device, and advance to the next iteration.
5. If the host device has battery power, unplug it and repeat **Step 3** and **Step 4** for the second iteration.

Common error messages

- None.

Passing criteria

- 2/2 (100%) iterations must pass in order to complete with passing status.

Input Resolution (touchpad-input-resolution)

Article • 01/24/2024

This is to validate that the Windows Precision Touchpad device is able to report at its specified resolution.

Test name

- Test.InputResolution.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.InputResolution

Test purpose

- Verifies that the device is able to genuinely report at the resolution that it specifies, and that resolution is greater than or equal to 300DPI.

Tools (Required)

- PTLogo.exe

Tools (Optional)

- A 7mm diameter capacitive contact.
- Robotic contact testing rig (recommended – task can be difficult if performed manually). Contact your robot vendor for details on operation.

Validation steps

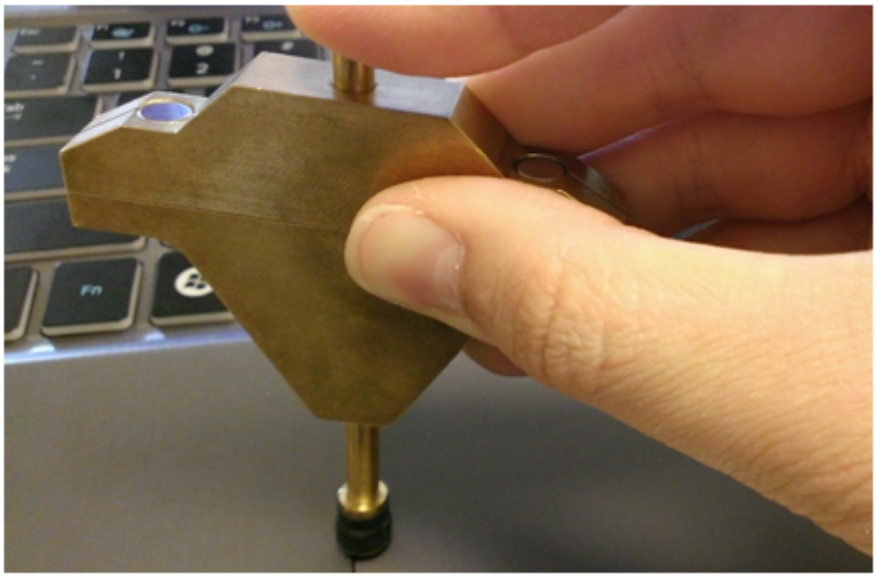
Between 2 slow horizontal swipes, every logical unit of X should be hit and reported, and between 2 slow vertical swipes, every logical unit of Y should be hit and reported.

1. Launch Test.InputResolution.json.
2. Starting from the left edge of the digitizer surface, drag a single contact across the touchpad at a velocity inversely proportional to the device's reported input resolution for the x axis from left to right, and again from right to left. See the following bullet item on *Drag Velocity* for more information. a. *Drag Velocity*: All devices must report at greater than or equal to 125Hz for a single contact. So the

following formula may be used to calculate a safe drag velocity: Drag Velocity <= 60Hz / (Logical Range for Axis / Physical Range for Axis).

For example, for a device with a logical range of 4000 for X, and a physical range of 100mm for X, a safe drag velocity would be:

60 / (4000/100) = 60 / 40 = 1.5mm/s.



b. All off-axis movement or jitter is ignored in this test. As shown in Figure 10, optionally a precision contact may be used to overcome potential friction with a human finger, but it must be kept perfectly perpendicular to the touchpad surface so the contact size does not drop far below 7mm (which could cause the device to “lose” the contact). 3. Repeat vertically. Starting above the top edge of the digitizer surface, drag a single contact down the touchpad at a velocity inversely proportional to the device’s reported input resolution for the y axis, dragging first from top edge to bottom edge, then from bottom edge to top edge. See the preceding bullet item on *Drag Velocity* for more information.

Note If enough logical x/y coordinates are hit after one swipe on that axis, the iteration will pass automatically without a need to swipe in the other direction.

Common error messages

- "[20] positional delta too large: #" o There was a jump in coordinates between 2 subsequent reports that exceeds the permitted tolerance of 0.5mm. o The given value is the length of the jump. o This error will stop an iteration, even if only 1 swipe has been performed.
- "[35] logical coordinate not found: #" o Either the minimum or maximum logical value was never reported in either of the 2 passes for a given axis. o The value

given is the X or Y logical value which was not reported (respective to the axis being validated).

- "[31] Too low percent of logical coordinates found: #" o The minimum percentage (95%) of all logical units required to be reported was not met for a given axis. o The value indicates the percentage of all logical units that were reported for a given axis.

Passing criteria

- 2/2 (100%) iterations must pass in order to complete with passing status.

Latency - Panning (touchpad-latency---panning)

Article • 01/24/2024

This is to test the reporting capabilities of the Windows Precision Touchpad device, with regard to latency and panning.

Test name

- Test.StepMotor.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Latency

Test purpose

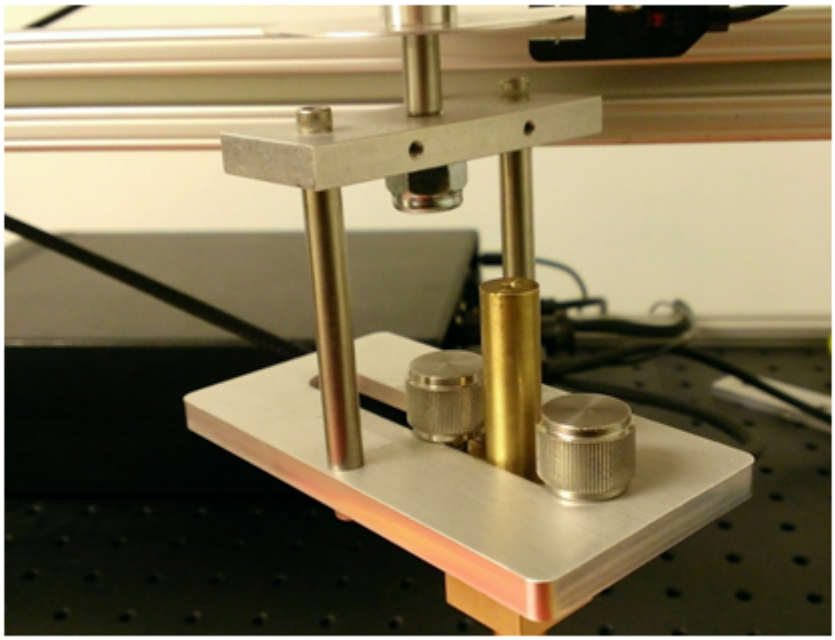
- Verifies that the device is able to report changes in the contact position, within the required latency window.

Tools required

- RA Tool with Precision Touchpad Modifications. For vendor information about the RA tool, see Hardware Requirements and Vendor Information.
- ptstepmotor.exe.

Validation steps

1. Make sure the RA tool is fitted with the smaller, adjustable PTP bracket as shown in the following image.



2. Plug the RA tool controller into the test machine, *and then* turn on the controller. Note that if you don't perform the tasks in this step in the correct order, the controller might end up in an undefined state.

The switches on the controller should initially be set as follows:

a. Motor: off b. Calibrate: Off c. Mode: Tachometer 3. Adjust the radius of the circle to 12mm and position the PTP device so that the contact will stay on the digitizer surface during its full rotation.

4. Rotate the arm until the notch on the disk is 2-3mm past the sensor.

a. If it is too close to the sensor, you will get an extra data point at the beginning of the test. This is because when the motor is started, it will jerk backwards. If the notch falls under the sensor during this, it will cause erroneous data.

b. If it is too far from the sensor, Count will seem to increment by 2, instead of giving 2 increments of 1. The motor is calibrated to rotate around and pass slowly for a certain segment of the circle relative to where it started, and that slow section is expected to pass under the sensor. But if it starts too far away from the sensor, the arm will still be moving too quickly when the notch passes under the sensor.

5. Run ptstepmotor.exe.

6. Lower the contact onto the touchpad surface.

7. Press the Spacebar on the test machine & flip the Motor switch to on to begin the Calibration phase.

8. After 9 rotations (when Count on the screen equals 17), the motor will come to a complete stop for 5 seconds. Press the Spacebar during those 5 seconds to advance to Acquisition.

9. Wait for 10 rotations of the arm, then press the Spacebar to move into the Collection phase.

10. After the arm has made 10 rotations in the Collection phase, press the Spacebar one last time to close the test.

Note that, unless there's an error, do not exit by pressing Esc. If you press Esc to exit when there is no error, the test may not finish.

For more detailed setup and test instructions, see How to measure Touch Panning Latency.

Common errors

- At the beginning of the Calibration stage, if the slot on the disk is too close to the photo interrupt signal, the calibration text will appear with a count of zero (0), before the arm has rotated once.

  Be sure to position the slot 2-3mm past the photo interrupt signal.

- When calibration executes correctly, the count should increment by 2 every rotation: once before and once after hitting the photo interrupt signal.

  o Because of this, the pause before beginning the Acquisition stage always occurs when the count = 17

  o If the count isn't incrementing properly, abort the test. Turn off the controller, unplug and re-plug the controller into the test machine via USB, then turn the controller on again. Start test from the beginning, and verify that count is incrementing correctly.

- At the end of the Calibration stage, the motor will pause completely for 5 seconds. During that pause, the you must press the Spacebar to advance to the Acquisition stage, otherwise an error message will be displayed.

Passing criteria

- If the test records an average panning latency of less than or equal to 70.0ms, the test will result in a pass.

Latency - Touch Down (touchpad-latency---touch-down)

Article • 01/24/2024

This is to test the reporting capabilities of the Windows Precision Touchpad device, with regard to latency and touch down.

Test name

- Test.AudioTouch.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Latency

Test purpose

- Verifies that the device is able to report the first contact down within the required latency window.

Tools required

- RA Tool. For vendor information about the RA tool, see Hardware Requirements and Vendor Information.
- ptaudiotouch.exe

Validation steps

For detailed setup and test instructions, see How to measure Touch Down Latency by using an Acoustic Measurement Tool. a. Note: use ptaudiotouch.exe, rather than the touch executable.

b. Only 50 taps during the collection phase are required for this class of device.

Common error messages

- None.

Passing criteria

- A recorded average touch-down latency of less than or equal to 35.0ms will result in a pass.

Linearity (touchpad-linearity)

Article • 01/24/2024

This is to test the linearity performance of a Windows Precision Touchpad device.

Test names

- Test.LinearityDiagonal
- Test.LinearityDiagonalMultiple

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.FingerSeparation
- Device.Input.Digitizer.PrecisionTouchpad.Jitter

Test purpose

- Verifies that the device is able to meet the linearity requirements in all directions, at the minimum contact separation distances.

  Note The mechanical pieces used for this test can be used to validate other user experience metrics by determining performance at narrower finger separations. While this additional test is not required, it is recommended.

Tools required

- PT3 with Precision Touchpad Modifications
- PTLogo.exe

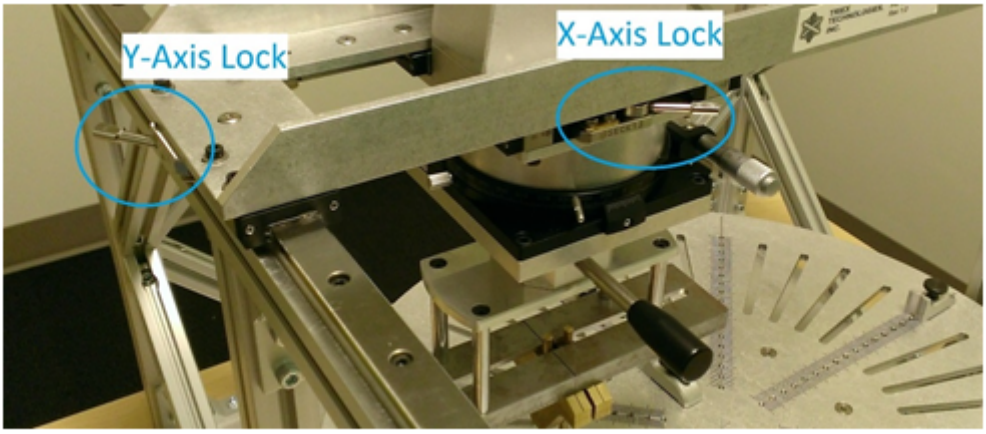
Validation steps

Note that the Linearity test is broken up into 6 sub tests. These tests can be performed in any order, and it is not necessary to follow the order in which they are presented here.

1. Launch Test.LinearityDiagonal.json.

a. Unlock the Y axis on the PT3, and unlock the X axis.

Here's an image showing the axis locking levers.



b. Partially rotate the plate to allow a corner-to-corner diagonal drag (or as close to 45 degrees as possible).

c. Starting outside one of the touchpad's corners, drag 1 precision contact to the opposite corner. Perform 5 diagonal iterations.

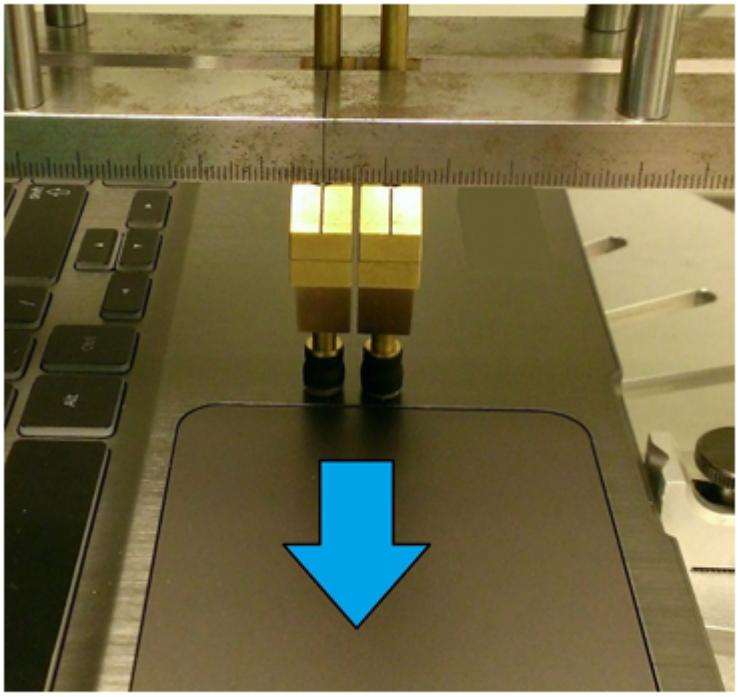
d. Be sure to alternate directions, and at some point rotate approximately 180 degrees to perform the opposite diagonal.

e. If no errors occur, PTLogo will automatically Pass the iteration and advance to the next.

2. Launch Test.LinearityDiagonalMultiple.json.

a. Add a 2nd 7mm precision contact to the PT3, with a center-to-center distance of 8mm between the two outer edges of the contacts (you may also set the 7mm contacts to 15mm center to center).

Here's an image showing the setup for the multiple contact test.



b. Rotate device to an angle (assuming it started at 0°). X axis should still be locked, with the Y axis unlocked.

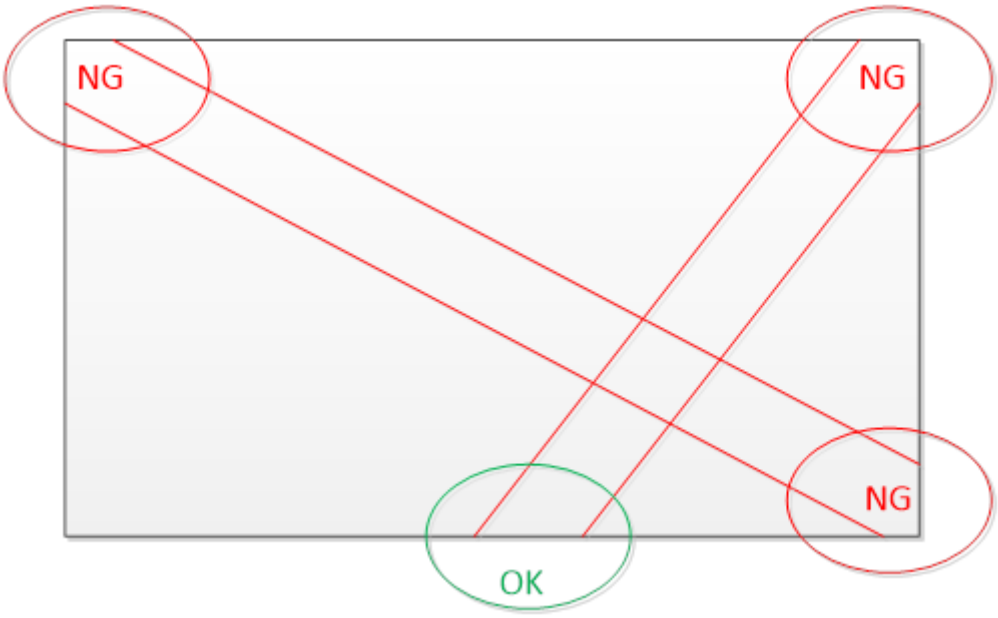
Here's an image showing the rotation guide on the PT3 test device.

c. Drag the 2 contacts across the touchpad (relative to the digitizer surface). Perform 3 diagonal iterations, alternating directions as shown in the following diagram.

d. Rotate device approximately 180 degrees, and repeat for the last 2 iterations.

Note that, as shown in the preceding diagram, all segments involve entry of both contacts on the same edge of the digitizer and exit of both contacts on the same edge of the digitizer.

Here's a diagram showing unacceptable segments for performing the multiple-contact diagonal test.



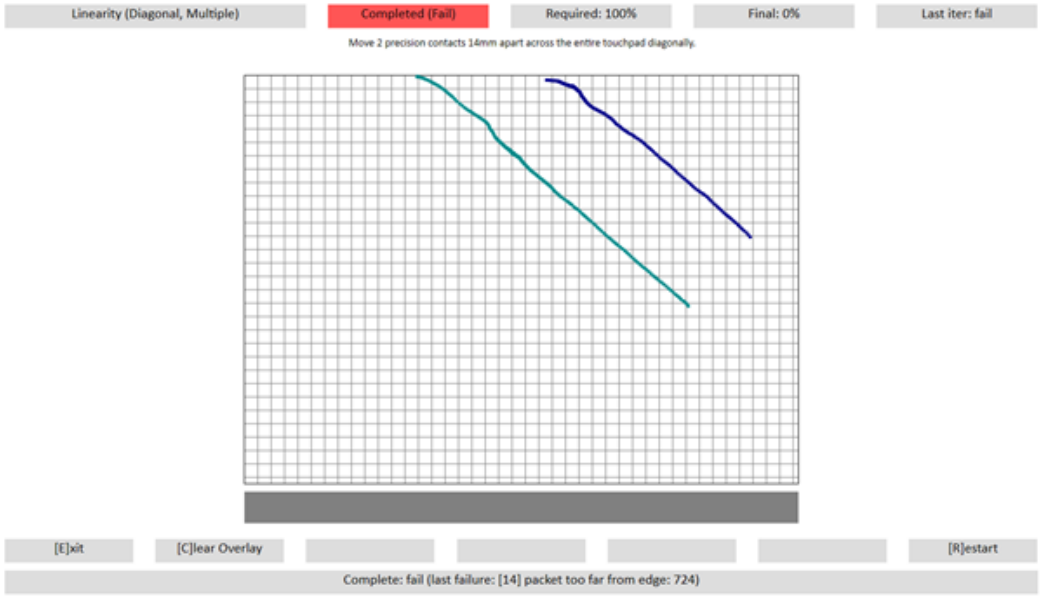
Note that, as shown in the preceding diagram, the Not Good (NG) segment portions involve contacts that did not enter the digitizer area from the same edge. As such 3 out of 4 of the segment entries are invalid.

Common error messages

- "[14] Packet too far from edge: #"

Swipe must start at the edge (or before it).

Here's a screenshot from the test, with a blue line showing a swipe that did not start *at, or before* the edge.



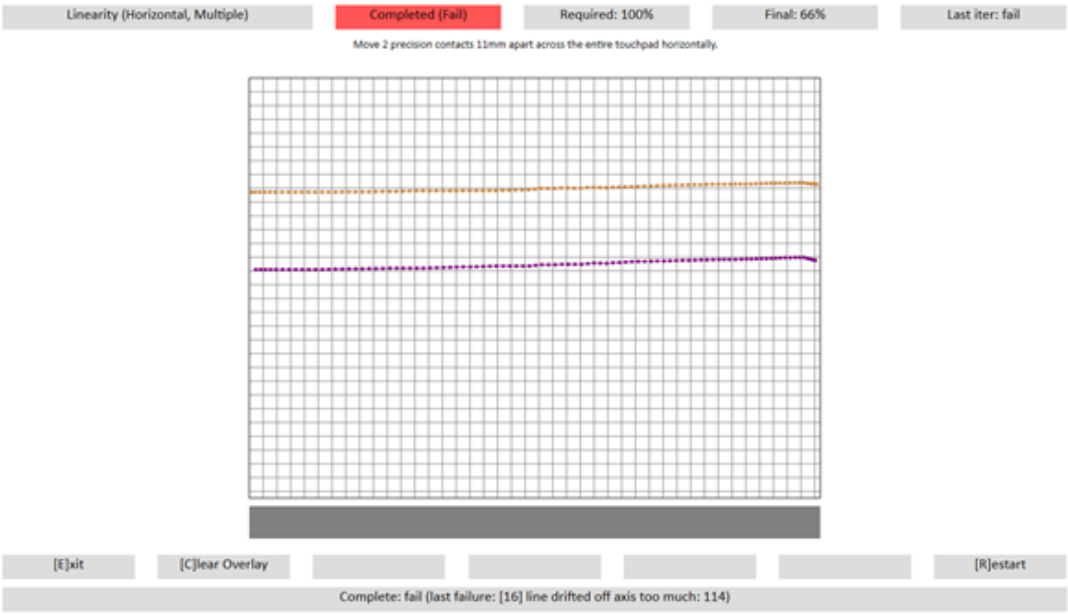
- "[15] Contact didn’t move in straight line."

  Reported path strayed from a straight line by more than the allowed tolerance (0.5mm in either direction).

- "[16] Line drifted off axis too much: #"

  The line moved in a (mostly) straight line, but was not completely horizontal, or completely vertical (the expected orientation is shown at the top of the screen). Unless caused by slight curves at the ends of the contact, this is usually a test error. Adjust the PT3 and try again.

  Use the visualization grid to help determine proper alignment. For example, in the following image, the device needs to be rotated counter-clockwise slightly.



- "[26] No packets outside border region."

  o This means that a contact never appeared more than a few millimeters away from the edges of the touchpad.

  o This can be caused by performing a horizontal swipe too close to the top of the touchpad, for example.

- "[27] Saw packet travel backwards at (#, #)."

  o Linearity tests are always performed by moving a contact in a consistent direction across the touchpad. Therefore, if a packet reports coordinates behind the previous packet, this is considered backwards jitter.

  o The displayed numbers (#, #) are the X and Y coordinates, in himetric units.

- "[34] Duplicate packet: #."

  o When contact is moving above a certain speed, each packet is expected to be at a different coordinate from the one before or the one after. So if two packets in such a situation report the exact same coordinates, they must be erroneous.

  o The displayed number (#) is the timestamp of one of the duplicate packets.

Passing criteria

- A total of 30/30 (100%) iterations must Pass in order to complete with passing status.

Maximum Supported Contacts (touchpad-maximum-supported-contacts)

Article • 01/24/2024

This is to test the maximum number of contacts supported by a Windows Precision Touchpad device.

Test name

- Test.MinMaxContacts.json

Core requirements tested

- Device.Digitizer.PrecisionTouchPad.Performance.MinMaxContacts

Test purpose

- Verifies that the device is able to support between 3 and 5 simultaneous contacts.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.MinMaxContacts.json.
2. PTLogo will automatically pass or fail depending on the maximum number of contacts the touchpad device supports.

Common error messages

- "[23] device doesn’t support minimum number of contacts:#"

  o The device supports less than three contact minimum.

  o The displayed value (#) indicates the number of supported contacts.

- "[24] device supports too many contacts:#"

  o The device supports more than a maximum of five contacts.

  o The displayed value (#) indicates the number of supported contacts.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Positional Accuracy (touchpad-positional-accuracy)

Article • 01/24/2024

This is to test the positional accuracy for reporting contacts on a Windows Precision Touchpad device.

Test name

- Test.PositionalAccuracy.json

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Accuracy

Test purpose

- Verifies that the device is able to report absolute position accurately.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.PositionalAccuracy.json.
2. Follow the directions on the screen to perform 3 taps in an indicated area on the touchpad. After each set of 3 taps, the instructions will ask you move to a different area, for a total of 15 taps. If instructions say to tap in a corner, tap as close to that corner as possible.

For each iteration, do the following:

a. Tap the location specified on the screen.

Note The specified locations are calculated as 13mm square areas. For example, if the instructions say "Top Left corner" you should tap somewhere within a 13mm square region of the top-left corner of the touchpad.

b. If no tap is registered by the device, manually fail the iteration.

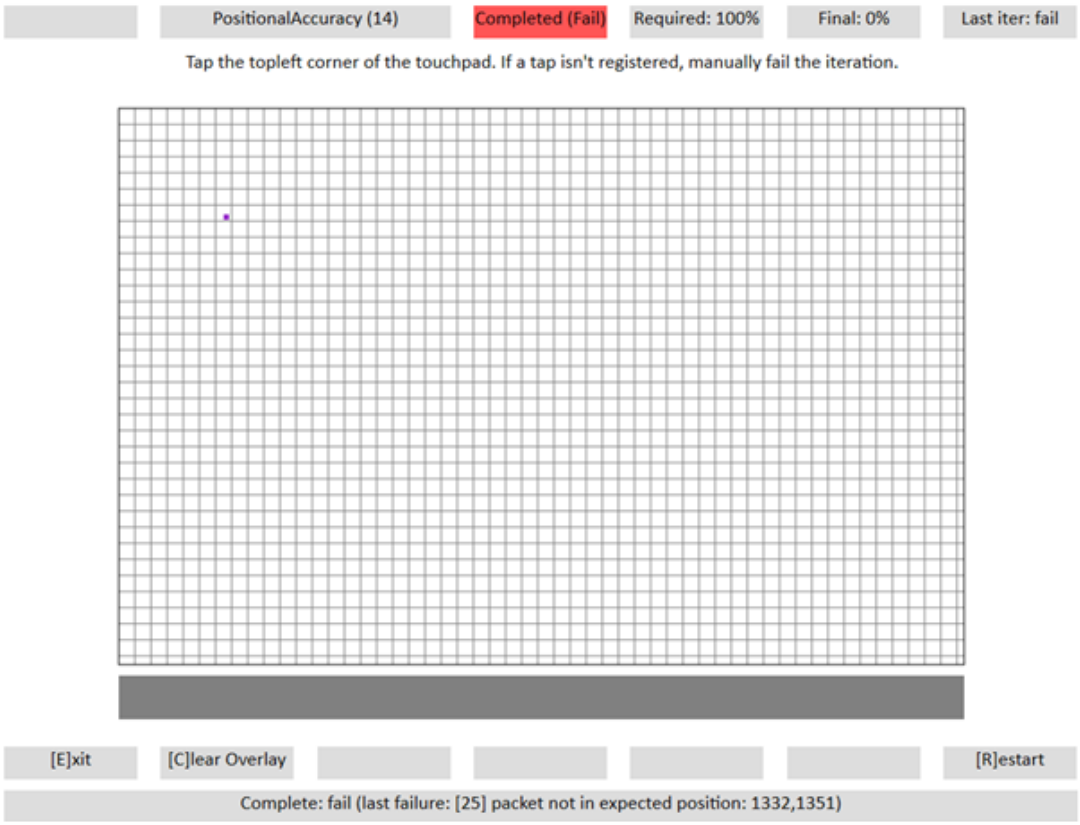
Common error messages

- "[25] packet not in expected position: XXX, YYY"

o Tap must occur at the location specified.

o The given values specify the logical coordinates of where the contact was reported.

Here's a screenshot showing a contact location that is outside the expected region.


PositionalAccuracy (14)
Completed (Fail)
Required: 100%
Final: 0%
Last iter: fail
Tap the topleft corner of the touchpad. If a tap isn't registered, manually fail the iteration.
[E]xit
[C]lear Overlay
[R]estart
Complete: fail (last failure: [25] packet not in expected position: 1332,1351)

Passing criteria

- 15/15 (100%) iterations must pass in order to complete with passing status.

Positional Accuracy - Manual (touchpad-positional-accuracy---manual)

Article • 01/24/2024

This is to test the positional reporting accuracy of a Windows Precision Touchpad device.

Test name

- Test.PositionalAccuracyManual.json

Core requirements tested

- Device.Input.Digitizer.Base.HIDCompliant

Test purpose

- Verifies that the device is able to report absolute position accurately.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.PositionalAccuracyManual.json.
2. **Iteration One:**

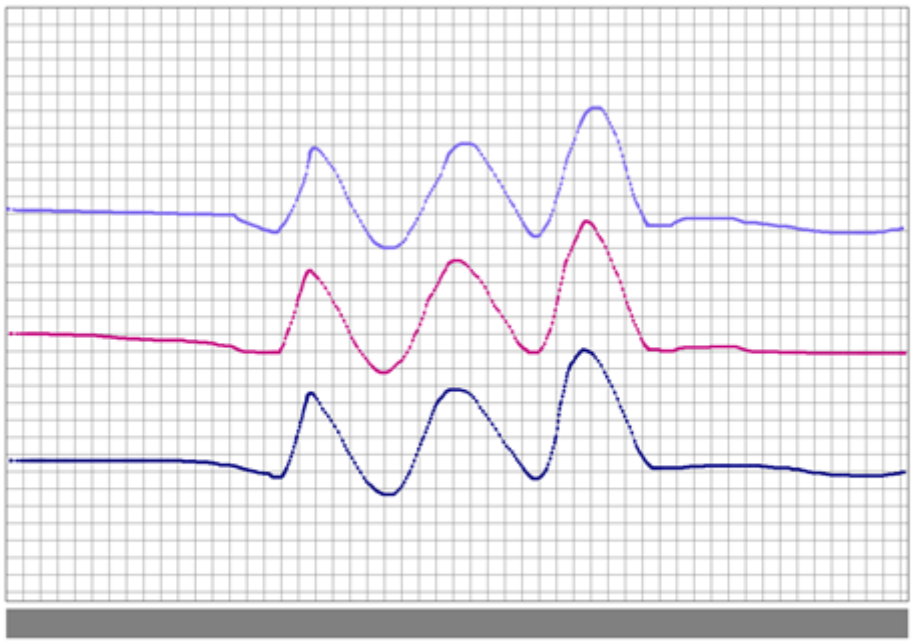
o Horizontally drag three fingers straight across 1/3 of touchpad.

o Without lifting, continue moving across the touchpad in a zigzag pattern.

o Without lifting, continue in a straight line across the last 1/3 of the touchpad.

o Verify that the displayed image matches physical finger movements, and then manually Pass or Fail the test accordingly.

Here's a screenshot from the test, showing traces from **Iteration One**.

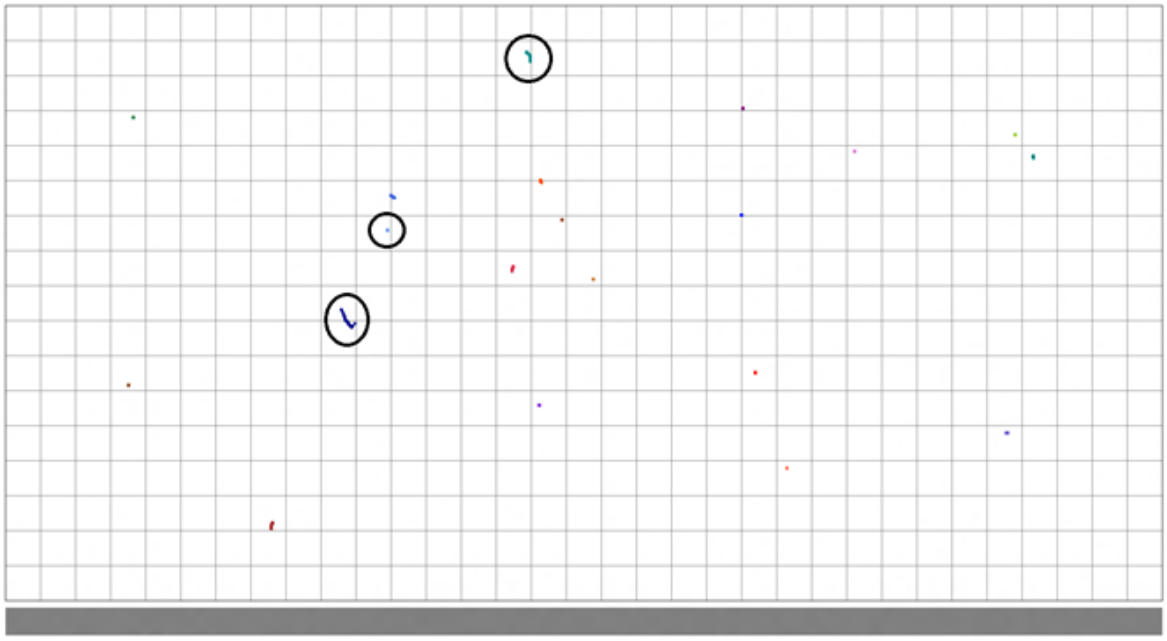


3. Repeat vertically for **Iteration Two**, and then diagonally for **Iteration Three**.

4. **Iteration Four:**

o Tap 20 times randomly on the touchpad.

o Verify that the displayed image matches your taps and there is no jitter, as shown in the following screenshot. Manually Pass or Fail the test accordingly.



In the preceding screenshot, the majority of taps are fine. The top circled contact is borderline – there was travel over the contact lifetime, but not a lot. Generally, up to

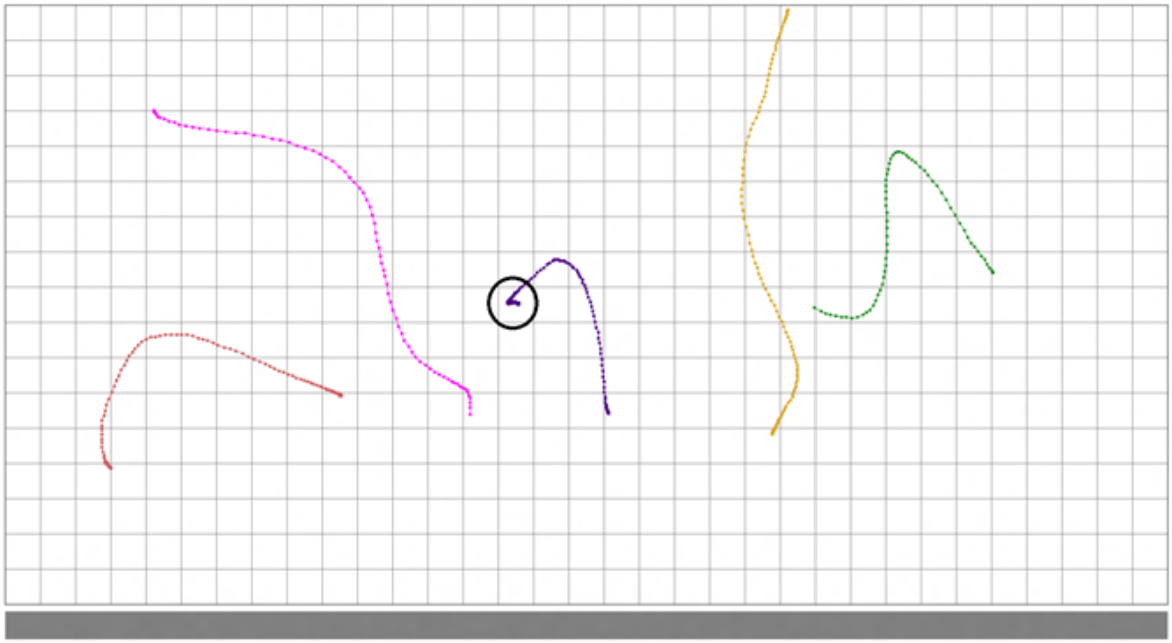
1mm (half a grid line) should be considered a Pass, as long as most are like the middle circled contact. The bottom circled contact shows travel greater than half a grid, which should be considered a failure for this test.

5. **Iteration Five:**

o Draw a curved line with your finger.

o Without slowing or changing direction, lift your finger.

o Verify that the displayed image matches your curves and there is no jitter on lift, as shown in the following screenshot. Manually Pass or Fail the test accordingly.



The key issues to look for in this iteration, are the hooks that are drawn when you lift your finger. In the circled example, the contact jittered to the left by a significant amount on contact lift. Consider any hook larger than a quarter of a grid as a failure.

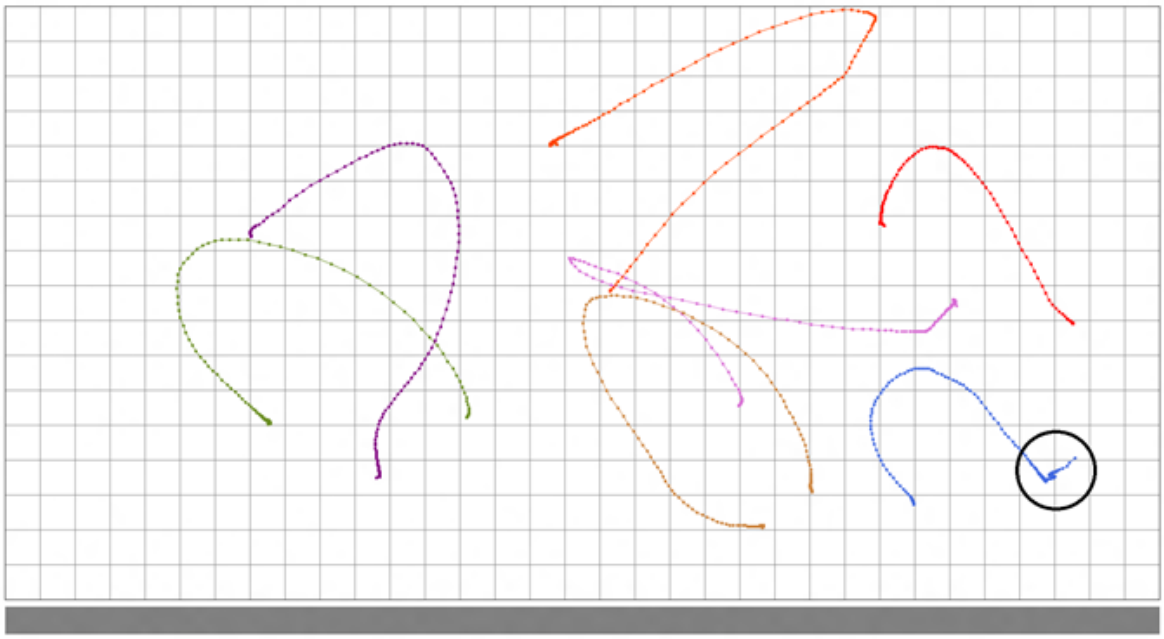
6. **Iteration Six:**

o Draw a curved line with your finger.

o Without lifting your finger, pause for 1 second.

o Click the physical button, and lift.

o Verify that the displayed image matches your curves and there is no jitter on lift, as shown in the following screenshot. Manually Pass or Fail the test accordingly.



In this test, there will be extra data as the physical button is clicked, so it's harder to judge by looking at the data itself. Instead, watch the mouse cursor as you click and lift. If the cursor moves more than half a grid box as you click, consider this test a failure. In the circled example, the cursor moved significantly on click and the example should be considered a failure.

Common error messages

- None.

Passing criteria

- 6/6 (100%) iterations must pass in order to complete with passing status.

Power State Transitions (touchpad-power-state-transitions)

Article • 01/24/2024

This is to test the behavior of a Windows Precision Touchpad device during power state transitions.

Test name

- Test.PowerStateReliability.json

Core requirements tested

- Device.Input.Digitizer.Base.ContactReports
- Device.Input.Digitizer.PrecisionTouchpad.Latency

Test purpose

- Verifies that the device is able to wake the system from sleep state S3, with both contact and button presses, and also handle power state transitions gracefully.

Tools required

- PTLogo.exe

Validation steps

1. Verify that the touchpad calibrates from a small contact on boot. Perform the following steps: a. Shut down the system.

b. While the system is powered off, place one small capacitive contact (either a finger or the PT3 device) on the digitizer surface.

c. Power on the system and leave the contact there until login is complete.

d. Lift contact, then interact with the PTP (move cursor, 1-finger tap, 2-finger tap, and 2 finger panning).

e. Verify that the touchpad behaves as expected.

2. Verify that the touchpad calibrates from a large contact on boot. Perform the following steps:

a. Shut down the system.

b. While the system is powered off, place an entire palm on the digitizer surface.

c. Power on the system and leave the contact there until login is complete.

d. Lift contact, then interact with the PTP (move cursor, 1-finger tap, 2-finger tap, and 2 finger panning).

e. Verify that the touchpad behaves as expected.

3. Verify that the touchpad powers up quickly. Perform the following steps:

a. Completely shut down the system, then boot it up again.

b. As soon as the logon screen comes up, start performing quick swipes on the touchpad.

c. Verify that the cursor moves.

4. Launch Test.PowerstateReliability.json.

5. If the touchpad behaved as expected in each of the preceding scenarios, manually pass the test. If not, fail the test.

Common error messages

- None.

Passing criteria

- 6/6 manual iterations must pass in order to complete with passing status.

PTPHQA (touchpad-ptphqa)

Article • 01/24/2024

This is to test the PTPHQA reporting capabilities of a Windows Precision Touchpad device.

Test name

- Test.PTPHQA

Core requirements tested

- Device.Input.PrecisionTouchpad.HIDCompliance.PTPHQA

Test purpose

- Verifies that the device is able to report a PTPHQA certification blob.
- Test only checks that the usage is available. For Windows 8.1 compatibility, the device is responsible for accurately reporting its PTPHQA blob.

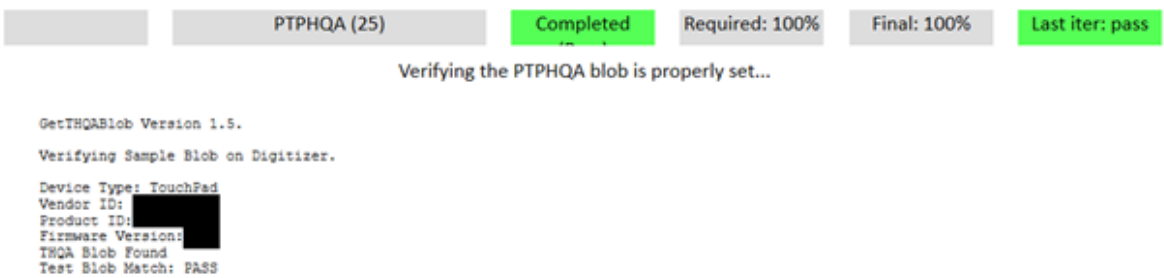
Tools required

- PTLogo.exe

Validation steps

1. Launch Test.PTPHQA.json.
2. PTLogo will automatically pass, if the device correctly reports the PTPHQA usage. Conversely, the test will fail if the PTPHQA usage is incorrectly reported.

Here's a screenshot from the test, showing a passing result.



Note If this is a re-certification, this test will fail and should be manually verified by entering the following command: **GetThqaBlob.exe –usage=TouchPad -verifyCertified**

Common error messages

- None.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Report Rate - Single Contact (touchpad-report-rate---single-contact)

Article • 01/24/2024

This is to test the single contact reporting capabilities of a Windows Precision Touchpad device.

Test name

- Test.ReportRate.json

Core requirements tested

- Device.Digitizer.PrecisionTouchPad.Performance.ReportRate

Test purpose

- Verifies that the device is able to report a single contact at the required report rate of 125 Hz for a single contact.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.ReportRate.json.
2. Following the instructions on the screen, move a single contact on the digitizer surface for 1, 5, or 20 seconds (a clock in the upper right corner shows how many seconds have passed since the contact has been down).
3. Repeat for a total of 11 times – Note that the length of time will change for some iterations, so please read and follow the on-screen instructions for each iteration.
4. PTLogo will fail an iteration and display an error on lift if the measured report rate is too low, or if it exceeds 250 Hz. Otherwise, the iteration will pass upon lift, and PTLogo will advance to the next iteration.

Common errors

- If an iteration continues after the contacts have been lifted, the device may not have reported the clearing of the tip switch correctly.

Passing criteria

- 11/11 (100%) iterations must pass in order to complete with passing status.

Report Rate - Multiple Contacts (touchpad-report-rate---multiple-contacts)

Article • 01/24/2024

This is to test the multiple contact reporting capabilities of a Windows Precision Touchpad device.

Test name

- Test.ReportRateMultiple.json

Core requirements tested

- Device.Digitizer.PrecisionTouchPad.Performance.ReportRate

Test purpose

- Verifies that the device is able to report the maximum number of contacts at the required report rate of 10 higher than the screen's refresh rate (this requirement will not exceed the 125hz).

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.ReportRate.json.
2. Following the instructions on the screen, move around the required number of contacts on the touchpad for 1, 5, or 20 seconds (a clock in the upper right corner shows how many seconds have passed since all contacts have been down).

Contacts should be kept at least 8mm apart edge to edge.

3. Repeat for a total of 11 times – Note that the length of time will change for some iterations, so please read and follow the on-screen instructions for each iteration.
4. PTLogo will fail an iteration, and display an error if the measured report rate is lower than required, or exceeds 250 Hz. Otherwise, the iteration will pass upon lift, and PTLogo will advance to the next iteration.

Common errors

- If an iteration continues after the contacts have been lifted, then be aware that the device might not have reported the clearing of the tip switch correctly.

Passing criteria

- 11/11 (100%) iterations must pass in order to complete with passing status.

Right Click Reliability (touchpad-right-click-reliability)

Article • 01/24/2024

This is to test the right-click detection capabilities of a Windows Precision Touchpad device.

Test name

- Test.RightClickReliability.json

Core requirements tested

- No requirement – recommended for experience testing on click-pads and pressure-pads.

Test purpose

- Provides testing for the reliability of right clicks with contacts that are in the far corner of the digitizer and very quick button presses

Tools required

- PTLogo.exe

Validation steps

1. Launch Test. RightClickReliability.json.
2. Test the far corner's reliability:

a. Place a finger in the far bottom-right corner of the touchpad.

b. Press the touchpad button. The test will pass, if a right-click is recognized.

c. Release the contact.

c. Repeat once - for a total of two total iterations.

3. Test the device's responsiveness to quick presses:

a. Quickly press and lift a contact in the bottom-right corner of the touchpad, such that a button click occurs. The test will pass, if a right-click is recognized.

b. Repeat two more times - for a total of three total iterations.

Common errors

- If the far corner test fails, the device may report left clicks when a user is attempting to produce a right click in the region.
- If the quick press test fails, the device may report left clicks when a user quickly attempts to right click.

Passing criteria

- 5/5 (100%) iterations must pass in order to complete with passing status.

Selective Reporting (touchpad-selective-reporting)

Article • 01/24/2024

This is to test the selective reporting capabilities of a Windows Precision Touchpad device.

Test name

- Test.SelectiveReporting

Core requirements tested

- Device.Digitizer.PrecisionTouchPad.HIDCompliance.SelectiveReporting

Test purpose

- Verifies that the device is able to correctly respond to selective reporting requests from the host.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.SelectiveReporting.json as an Administrator (elevated).
2. Move 1 contact on the touchpad, and exert activation force.
3. Manually Pass the iteration, if the following is true (otherwise, manually Fail the iteration): then manually pass the iteration, otherwise manually fail the iteration.

The touchpad *does not* respond (the mouse cursor does not move, no contacts are displayed in PTLogo, and button clicks do not turn the on-screen bar green). **Common error messages**

- None.

Passing criteria

- 1/1 (100%) iterations must Pass in order to complete with passing status.

Static Validation (touchpad-static-validation)

Article • 01/24/2024

This is for performing static validation tests.

Test name

- Test.StaticValidation.json

Core requirements tested

- Device.Input.Digitizer.Base.ThirdPartyDrivers

Test purpose

- Verifies that device meets the fundamental bus interface requirements with the host.

Tools required

- PTLogo.exe

Validation steps

1. Launch Test.StaticValidation.json
2. PTLogo will run static validation test code, and verify static HID requirements.

Common error messages

- Error

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Stationary Jitter (touchpad-stationary-jitter)

Article • 01/24/2024

This is to test the stationary contact reporting abilities of a Windows Precision Touchpad device.

Test names

- Test.StationaryJitter
- Test.StationaryJitterMultiple

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.Jitter

Test purpose

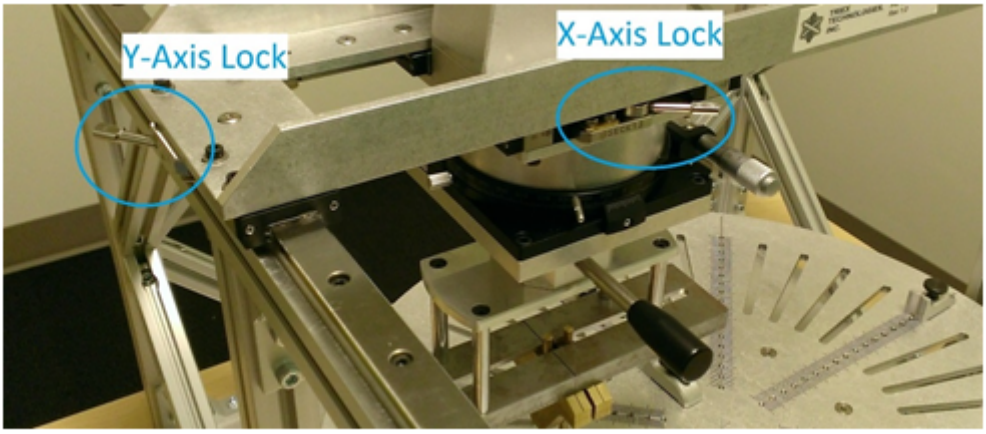
- Verifies that the precision touchpad is able to accurately report a stationary contact, or multiple stationary contacts for a period of 10 seconds.

Tools required

- PT3 w/Precision Touchpad Modifications
- PTLogo.exe

Validation steps

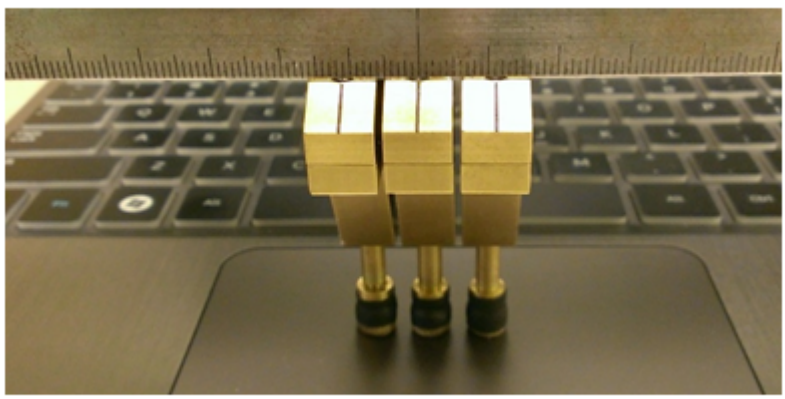
1. Launch Test.StationaryJitter.json.
2. Make sure both axes are locked on the PT3, so that the contact (or contacts) can only move up and down. The following image shows the PT3 and its axis locking levers.



3. Carefully lower 1 contact straight down anywhere on the digitizer surface.

Note Some PT3 tips might bounce slightly if brought down quickly. To avoid this, lower the tip slowly onto the digitizer surface.

4. Wait 10 seconds, using the timer in the upper-right corner of the PTLogo user interface.
5. Carefully raise the contact.
6. If no jitter is detected, PTLogo will Pass the iteration and move on to the next. Otherwise it will Fail the test and display an error.
7. Repeat for 5 iterations, each at a different location on the touchpad.
8. Launch Test.StationaryJitterMultiple.
9. Add contacts to the PT3, so that there are 3 to 5 contacts (as supported by the device) at a separation distance of 13mm edge to edge, as shown in the following image.



10. Carefully lower the contacts down anywhere on the touchpad, until all 3 contacts are visualized in PTLogo.

For pressure pads, lower the contacts until they are just above the maximum z height (1mm), and bring down quickly but smoothly until all 3 contacts are visualized in PTLogo.

11. Wait 10 seconds, using the timer in the upper-right corner of PTLogo.
12. Carefully raise the contacts.
13. If no jitter is detected, PTLogo will Pass the iteration and move on to the next. Otherwise it will fail the test and display an error.
14. Repeat for 5 iterations, each at a different location on the touchpad, but always with the contacts aligned horizontally on the digitizer surface as shown in the preceding image.
15. A second set of iterations will begin to test jitter after motion. Place 3 contacts on the touchpad and quickly move them at least 2cm, then let them sit. This must be performed within 3 seconds.
16. Wait 10 seconds for the test to validate, then raise the contacts.
17. Repeat for 5 iterations, each at a different location on the touchpad.

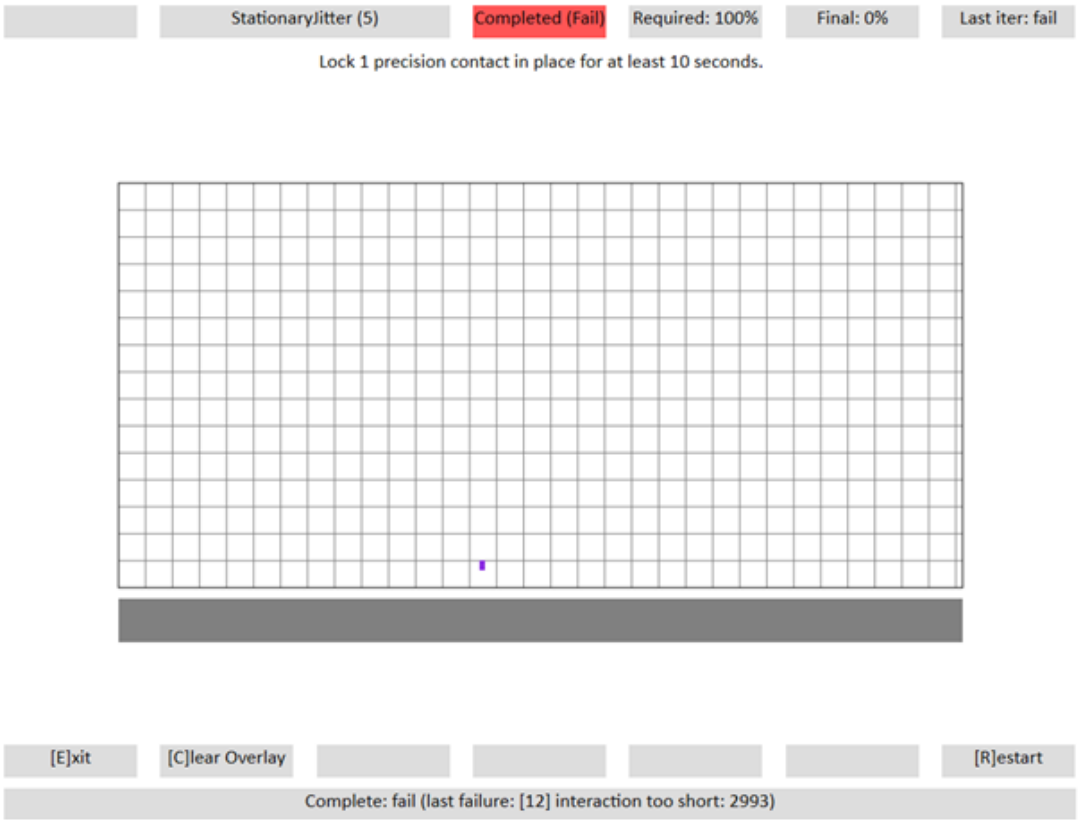
Common error messages

- "[12] interaction too short: #####"

  o Contact was down for less than the required duration.

  o The displayed value (#####), is the duration of the contact in milliseconds.

  Here's a screenshot from the test, showing an interaction that was too short.

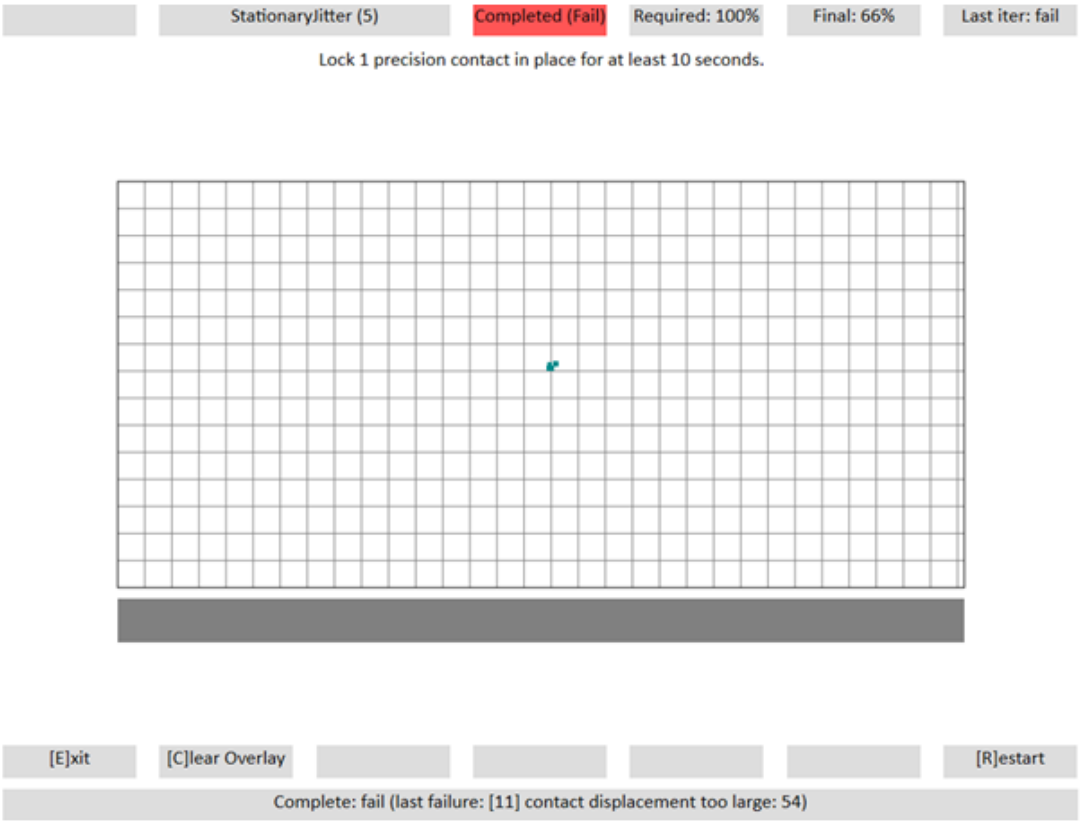


- "[11] contact displacement too large: ###"

  o The contact moved or jittered greater than the allowed tolerance of 0.5mm.

  o The displayed value (###) is the himetric displacement.

  Here's a screenshot from the test, showing a displacement that is too large.



Passing criteria

- A total of 10/10 (100%) iterations must pass in order to complete with passing status.

Z-Height Detection (touchpad-z-height-detection)

Article • 01/24/2024

This is to test the z-axis reporting behavior of a Windows Precision Touchpad device.

Test name

- Test.ZHeight

Core requirements tested

- Device.Input.Digitizer.PrecisionTouchpad.ContactTipSwitchHeight

Test purpose

- Verifies that the device does not report contacts that exceed the allowable detection threshold in the z-axis.

Tools required

- PT3 w/Precision Touchpad Modifications
- PTLogo.exe
- 1mm thick non-conductive material (such as a business card)

Validation steps

1. Place 1mm thick material on top of touchpad.
2. Lower 1 PT3 contact until it just touches the non-conductive material.
3. Remove the material, leaving PT3 contact 1mm above the touchpad.
4. Launch Test.ZHeight.json.
5. Leave contact in place above touchpad for 1 minute.
6. Verify that no contact is reported – and if that is the case, then manually Pass the iteration. Otherwise manually Fail the iteration.

Common error messages

- None.

Passing criteria

- 1/1 (100%) iterations must pass in order to complete with passing status.

Appendix (touchpad-appendix)

Article • 01/24/2024

The appendix for the Windows Precision Touchpad Validation Guide, provides information about common error messages, along with some HID-specific ones.

Error messages

Expand table

Error message #	Error message	Explanation
1	Explicitly failed by user.	Operator failed the iteration/test via hotkey.
2	Failed too many iterations:	Too many iterations failed. Includes the number of failed iterations.
3	Received input at an unexpected time.	The test did not expect to receive data, but data was received. For example, this can happen in the Z-Height Detection test.
5	Received too many contacts:	Operator placed too few contacts down. Includes the number of detected contacts.
6	Received too few contacts:	Operator placed too many contacts down. Includes the number of detected contacts.
7	Test didn't receive enough data.	If the error occurs on contact lift, then it indicates that the test required the contacts to last a certain duration, but that duration was not met. If it occurs on contact down, then indicates that the test received a zero-contact frame as the first data – this can be caused by a non-capacitive button press, but is more likely

Error message #	Error message	Explanation
		a protocol error (device sending an empty frame).
9	Received unexpected contact.	On a test that requires a specific number of contacts, this indicates that a contact came down after a contact went up. All contact-downs should occur before all contact-ups.
10	Geometry data outside expected range:	Only on **Test.Geometry**. Width and/or height were outside the range expected by the test. Includes the detected width and height in himetric units.
11	Contact displacement too large:	The contact's overall x/y displacement was too large. Includes the detected displacement in himetric units.
12	Interaction too short:	The interaction length (the time from first contact down to last contact up) was too short. Includes the detected interaction length in milliseconds.
13	Interaction too long:	The interaction length (the time from first contact down to last contact up) was too long. Includes the detected interaction length in milliseconds.
14	Packet too far from edge:	Indicates that the first packet was too far from the edge of the touchpad. Includes the distance from the edge in himetric units.
15	Contact didn't move in straight line:	The contact deviated too much from a line of best fit. Includes the maximum detected deviation from the line in himetric units.

Error message #	Error message	Explanation
16	Line drifted off axis too much:	Only applies to the **Test.Linearity** family of tests. The contact's displacement in either X or Y was too large. Includes the deviation in himetric units.
17	Not enough separation of points:	Only applies to the **Test.ConvergeDiverge** family of tests. If converging, then the beginning points were too close to each other. If diverging, then the ending points were too close to each other.
18	Too much separation of points:	Only applies to the **Test.ConvergeDiverge** family of tests. If converging, then the ending points were too close to each other. If diverging, then the beginning points were too close to each other. Includes the distance between points in himetric units.
20	Positional delta too large:	The position delta between two packets was too large for the test. Includes the detected delta in himetric units.
23	Device doesn't support minimum number of contacts:	The device doesn't support the required minimum number of contacts. Includes the number of supported contacts.
24	Device supports too many contacts:	The device supports more than the required maximum number of contacts. Includes the number of supported contacts.
25	Packet not in expected position:	Only applies to the **Test.PositionalAccuracy** family of tests. Indicates that the packet's location was not in the required position. Includes

Error message #	Error message	Explanation
		the packet's location in himetric units.
26	No packets outside border region.	On tests that involve drawing a straight line with the precision contact rig, this indicates that the entire line was in the border region of the touchpad.
27	Saw packet travel backwards.	Only applies to the **Test.Linearity** family of tests. Indicates that a packet was seen traveling backwards in relation to the rest of the packet stream.
28	Too low DPI:	Only on **Test.InputResolution**. Indicates that the logical range of X/Y on the touchpad, combined with its physical dimensions, do not support the required DPI. Includes the calculated DPI.
29	Saw confidence bit set after cleared.	Only on **Test.Confidence**. Indicates that a contact was seen setting the confidence bit after it had been cleared for that contact.
30	Confidence bit set too long:	Only on **Test.Confidence**. Indicates that the confidence bit was not cleared early enough in the contacts lifetime. Includes the length of time the confidence bit was set in milliseconds.
31	Too low percent of logical coordinates found:	Only on **Test.InputResolution**. The X or Y coordinate in packets received in a given iteration must include a minimum percent of the total range. Includes the percent actually found.
32	You must run this test elevated.	PTLogo should have been launched elevated for this test.

Error message #	Error message	Explanation
33	Device does not support selective reporting.	Device does not support selective reporting.
34	Duplicate packets:	Only applies to the **Test.Linearity** family of tests. Indicates that two consecutive packets had the same X/Y location, even though the contact was moving. Includes the scan time of the detected duplicate packet.
35	Logical coordinate not found:	Only on **Test.InputResolution**. Indicates that a required X or Y coordinate was never reported by any packet during the iteration. Includes the required coordinate in logical units.
36	Confidence always set.	Only on **Test.Confidence**. Indicates that the confidence bit was never cleared.
37	Failed via automation.	The test was failed via automation.
38	Contact displacement too small.	The contact's initial x/y displacement was too small. Includes the detected displacement in himetric units.
39	Unexpected button state change.	Only on **Test.RightClickReliability**. Occurs when a button-only input report is received from the touchpad.
40	Device height too small.	Only on **Test.DeviceHeight**. The touchpad's height is too small. Includes the detected height in millimeters.
41	Device width too small.	Only on **Test.DeviceWidth**. The touchpad's width is too small. Includes the detected width in millimeters.

HID-Specific Error Messages

Here's a table of the HID-specific error messages related to the Windows Precision Touchpad device tests.

Expand table

HID error #	Error message	Explanation
1	Invalid X bit size.	The bit count for Tx/Cx is outside the range [1,32]. Only checked if C is present.
2	Invalid Y bit size.	The bit count for Ty/Cy is outside the range [1,32]. Only checked if C is present.
3	Invalid packet transition.	This error message can be presented along with any of the following others: o "Last move location different" The coordinates of the 'tip switch clear' report for a given contact are not the same as the coordinates of the 'last tip switch set' report. o "Missing tip-on" The first report didn't have the tip switch set, or there were two packets in a row without the tip switch set. o "Missing tip" A contact present in the previously-reported frame, with the tip switch set, was not found in the current frame.
4	Invalid scan time.	This error message can be presented along with any of the following others: o "(Not present)" The device does not support the scan time usage in its descriptor.

HID error #	Error message	Explanation
		o "(Range)" The scan time reported is outside the logical range.. o "(Delta > 10ms more than 1% of the time)" The delta in scan time from frame to frame exceeds 10ms more than 1% of the time. o "(Delta > 16.7ms)" The delta in scan time from frame to frame was larger than 16.7ms. o "(Duplicate)" The scan time was duplicated in two sequential frames. o "(Differing values in frame)" The scan time value was not identical for all reported contacts of a given frame. o "(Drifted from wall clock)" The deltas in scan time drifted too far from system time. Scan time deltas were greater than 5% of clock time.
5	Invalid X, Y.	This error message can be presented along with any of the following others: o "(Invalid T)" With C, Width, or Height present, Tx and/or Ty were not present, or not within their logical range. o "(Invalid C)" With T, Width, or Height present, Cx and/or Cy were not present, or not within their logical range. o "(Invalid T/C combo)"

HID error #	Error message	Explanation
		T was not contained within bounding box formed by C, Width, and Height.
6	Invalid width or height.	This error message can be presented along with any of the following others: o "" Width and/or height was present, and either one was not present, or one/both were outside their logical range. o "(0)" Width and/or height were present, but the logical value for one/both was zero.
8	Invalid contact ID.	This error message can be presented along with any of the following others: o "(Not present)" The device does not support the ContactID usage in its descriptor. o "(Dupe in frame)" A contact ID was duplicated in a single frame (sometimes caused by an incomplete frame being reported).
14	No data in frame.	There were no contacts in the frame, and the physical button is not down, but the physical button was not previously up.
16	Invalid # of contacts in frame.	The number of contacts in the frame did not match the reported Actual Count.
17	More than max contacts in frame.	The number of contacts in the frame exceeded the maximum number of contacts the device supports per MAX COUNT.
18	Sampling rate out of range.	The sampling rate was not in the allowed range for the

HID error #	Error message	Explanation
		number of contacts reported.
21	Invalid actual count.	The device does not support the **ActualCount** usage in its descriptor.
22	Invalid confidence.	The confidence switch was not set (and the test was not the confidence test).

PTLogo Command Line Switches

These switches can be combined and are useful for debugging purposes only. These switches are not permissible for a certification test run.

Expand table

Switch	Usage	Purpose
-startat #	"Ptlogo.exe –startat # test.foo.json" (where # indications the specific iteration to start at for a given .json test)	Used to skip to a specific iteration in a given test.
-NoHIDValidation	"Ptlogo.exe –noHidValidation test.foo.json"	Used to disable background HID validation for a specific test.
-noDesktop	"Ptlogo.exe –noDesktop test.foo.json"	Used to start the test on the same input desktop from where it was launched; useful for running digiinfo in the background or other debugging tools.
-alliters	"Ptlogo.exe –alliters test.foo.json"	Used to allow the operator to go through all iterations of a test even if more than the maximum # of permissible failed iterations have occurred
-EnableHIDLogging	"Ptlogo.exe –enableHIDLogging test.foo.json"	Used to enable HID logging during a specific test to generate a HID.log file for detailed debugging of failures.

Touchscreen Devices

Article • 01/18/2019

This section provides information about how to implement and validate an integrated Windows Touchscreen device, for Windows 10 and later operating systems.

In this section

Topic	Description
Touchscreen Implementation Guide	This section is an implementation guide for an integrated Windows Touchscreen device, for Windows 10 and later operating systems.
Touchscreen Validation Guide	This section is a validation guide for a Windows Touchscreen device, for Windows 10 and later operating systems.

Touchscreen Implementation Guide

Article • 05/20/2024

This section is an implementation guide for an integrated Windows Touchscreen device, for Windows 10 and later operating systems.

The information includes guidance on using the Human Interface Device (HID) protocol to communicate with a Windows host.

In this section

Expand table

Topic	Description
Device Bus Connectivity	This topic discusses bus connectivity methods for an integrated Windows touchscreen device.
Power Management	This topic provides information about Touchscreen power management, and discusses power consumption requirements.
Protocol Implementation	This section provides guidelines for Touchscreen protocol implementation. Windows Touchscreen devices are expected to use the Human Interface Device (HID) protocol to communicate with the host.
Wake-on-Touch Implementation Guide	This topic provides information about how to implement wake-on-touch functionality.
Digitizer Display Mapping	This section provides guidelines for successfully mapping digitizers to their display.

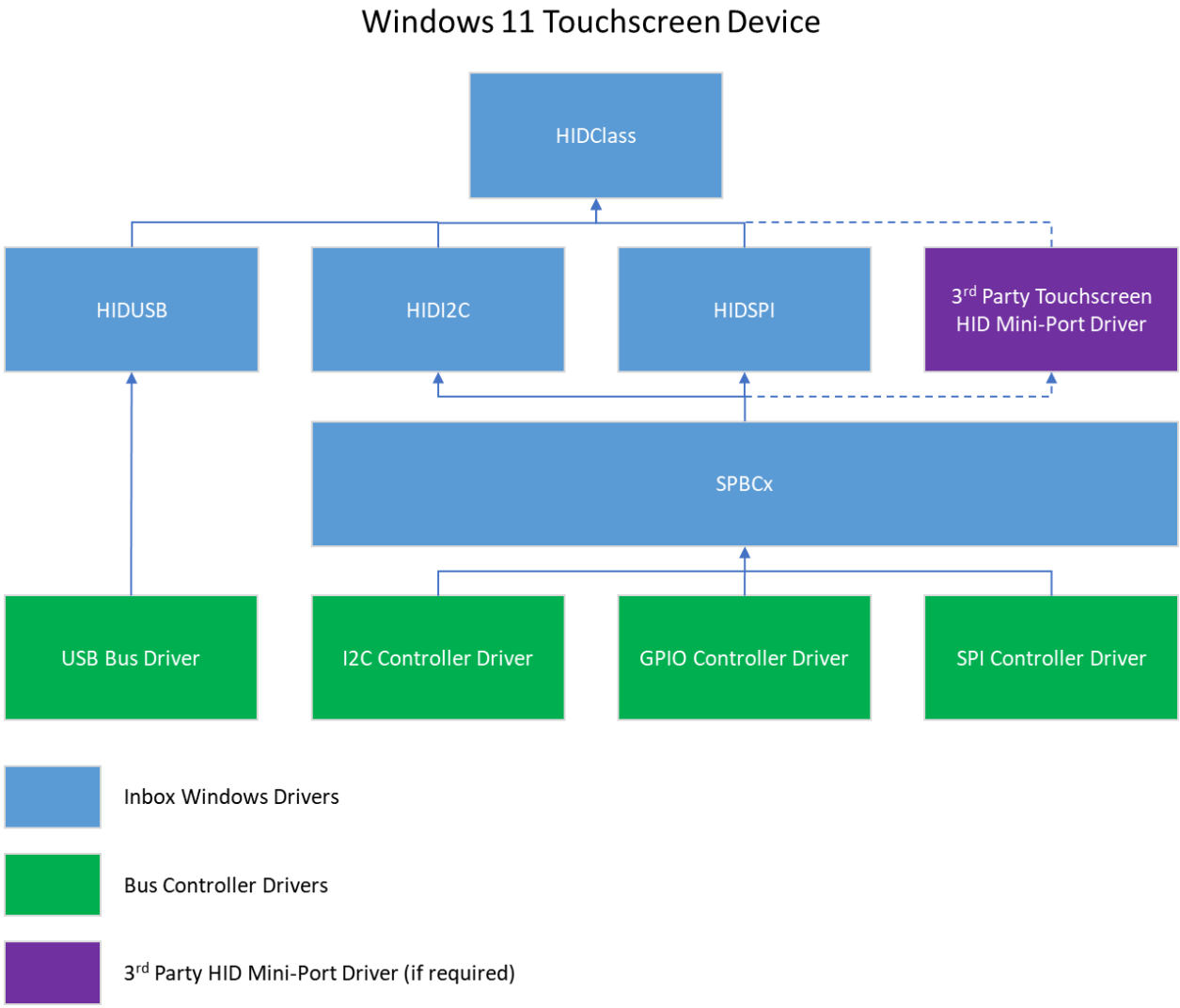
Device Bus Connectivity (touchscreen-device-bus-connectivity)

Article • 06/21/2022

> **Note**
>
> This article contains references to the term slave, a term that Microsoft no longer uses. When the term is removed from the software, we'll remove it from this article.

This topic discusses bus connectivity methods for an integrated Windows touchscreen device.

An integrated Windows touchscreen device can use the Microsoft-provided inbox drivers to connect to its Windows host, using either HID over USB, HID over I^2C, or (starting in Windows 11) HID over SPI. However you can use any other bus that you want, as long as you provide the required Windows-compatible, 3rd party HID mini-port driver for the touchscreen device. The following diagram shows the Windows 10 driver stack for a Windows touchscreen device.



Note that for full compatibility with Windows 11 for desktop editions (Home, Pro, Enterprise, and Education), we recommend using the Windows inbox drivers whenever possible. And if you have to use a 3rd party mini-port driver, then you must add this 3rd party driver to the appropriate OEM, and System Restore images, and then make these images available for download on Windows Update.

The following sections present some examples of device configurations.

I²C Devices

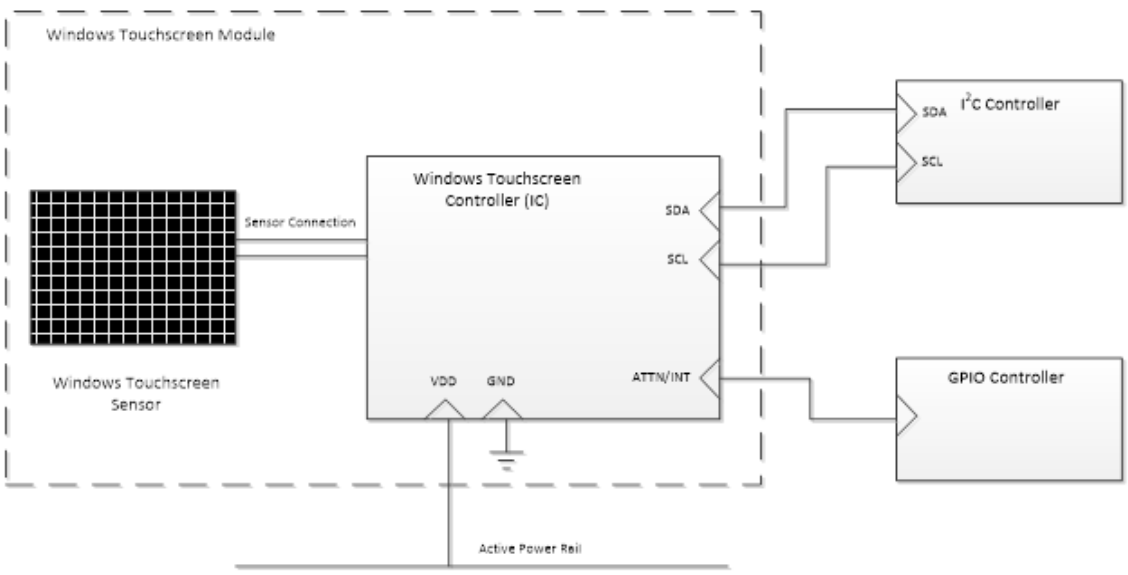
A Windows Touchscreen module is defined as the combination of a controller IC, a sensor, and any associated mechanisms.

A Windows Touchscreen module that connects to its Windows host via the I²C bus must, at a minimum, expose the following five connection pins:

- A data line (SDA)
- A clock line (SCL)
- An interrupt line

- A power supply line
- A ground connection (GND)

The following is a diagram of the connections lines between a Windows Touchscreen device and its Windows host.



When connecting to an I²C controller, it is important to understand the bandwidth demands of all the components sharing that controller. The minimum I²C clock speed of 400 KHz is recommended for a Windows Touchscreen device.

We recommend connecting the interrupt line (also referred to as an ATTN line) to an On-SoC GPIO controller, or to an IOAPIC. The GPIO or IOAPIC resource to which the interrupt line is connected, should be capable of (and configured for) waking the SoC. The wake-up capability allows the Windows Touchscreen to wake the system in various scenarios.

If you decide to use the wake functionality, the power line that is connected to the Windows Touchscreen device should not be shared with other devices which are not wake-capable. In order for wake scenarios to function properly, the power line that is used must be energized during connected standby/S3 conditions.

ACPI Table Entries

A Windows Touchscreen device that is connected via I²C must define an entry in the Advanced Configuration and Power Interface (ACPI) table of the host, for the device to be recognized by the host. For more information about ACPI, see Advanced Configuration and Power Interface Specification.

The ACPI table entry should specify the following information:

Entry	Description
ACPI Device Entry Name	A 4-character identification unique to the ACPI table, to reference the device. For example, “TOUC”.
ACPI Hardware ID	A 4-character + 4-number ACPI hardware ID, to reference the device. This is exposed in device manager. For example, “MSFT0001”.
Compatible ID	This should always be “PNP0C50” to indicate that the device is HID I^2C compatible.
I^2C Controller	Specifies an I^2C controller on the Windows host. This controller is used to connect the Touchscreen to the Windows host, and makes it possible for the Touchscreen and the host to communicate. For example, “I2C3” – to indicate I^2C controller #3.
I^2C Slave Address	Specifies the I^2C slave address for the device. The host uses this address to single out the Touchscreen device on the I^2C bus for communication. For example, “0x6F”.
I^2C Speed	Specifies the maximum speed supported by both the device and the I^2C controller. Specifying the speed in the ACPI table ensures reliable communication. This speed should not be any lower than 400KHz (0x61A80).
GPIO Controller	The GPIO controller to which the interrupt line of the Touchscreen device is connected. This tells the host where to "listen" for interrupt signals. For example, “GPIO0” – to indicate GPIO controller #0.
GPIO Resource/Pin	The GPIO controller pin to which the interrupt line of the Touchscreen device is connected. The host then associates this specific GPIO pin with interrupt signals from the Touchscreen. For example, “{35}” – to indicate pin 35.
GPIO Resource Type	Defines the constraints around the GPIO resource. This entry for the ACPI table, should be set to “Exclusive” unless you want to select SoC wake. If you decide to select SoC Wake, then set this entry to “ExclusiveAndWake”.
GPIO Interrupt Assertion Type	Defines the type of triggering that the Touchscreen device will provide for its interrupts. This can either be "Edge-triggered", or "Level-triggered." HID I^2C-compliant devices should use “Level-triggered" interrupts.
GPIO Interrupt Assertion Level	Defines the voltage level on the interrupt line, when the interrupt is asserted by the device. This can be specified as “ActiveLow” or “ActiveHigh”.

USB Devices

A high-speed/full-speed integrated Windows Touchscreen device that is connected via USB 2.0, should expose the necessary pins for host connectivity.

Connection to the host can take many forms and is at the discretion of the integrator.

Note that, when connecting to a USB hub, it is important to understand the bandwidth demands of all the components that share the hub. It is highly recommended that high-bandwidth devices, and Windows Touchscreen controllers do not share the same USB hub, as this may result in bandwidth demands that exceed bus capability.

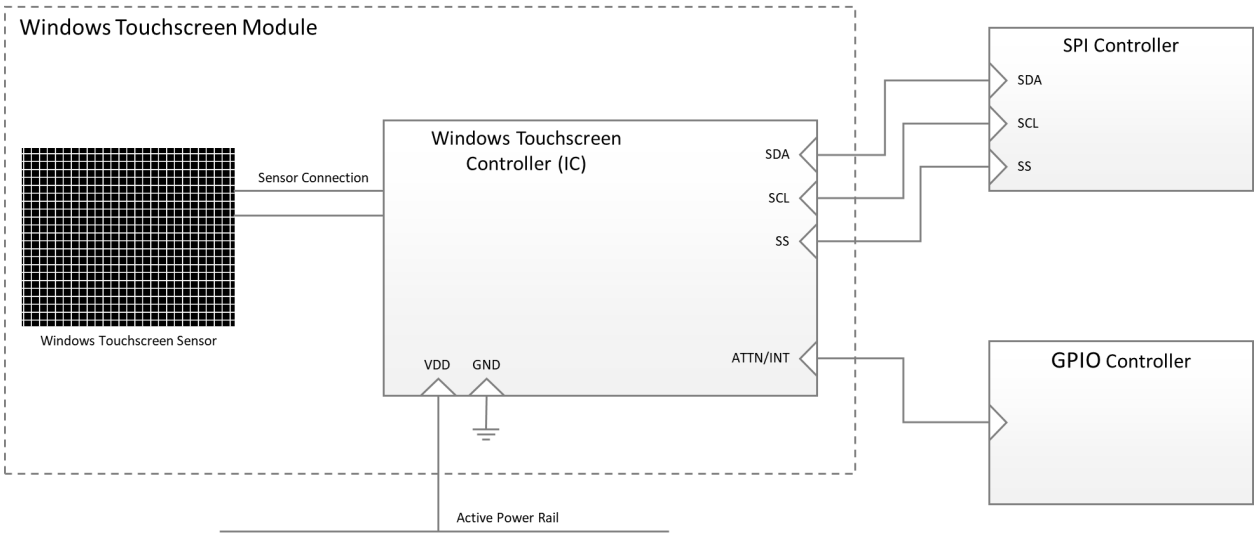
HID SPI Devices

An Windows Touchscreen module is defined as the combination of a controller IC and a sensor.

A Windows Touchscreen module that connects to its Windows host via the SPI bus must, at a minimum, expose the following five connection pins:

- A data line (SDA)
- A clock line (SCL)
- A chip select line (SS)
- An interrupt line
- A power supply line
- A ground connection (GND)

The following is a diagram of the connections lines between a Windows Touchscreen device and its Windows host.



ACPI Table Entries

A Windows Touchscreen device that is connected via SPI must define an entry in the Advanced Configuration and Power Interface (ACPI) table of the host, for the device to be recognized by the host. For more information about ACPI, see Advanced Configuration and Power Interface Specification.

The ACPI table entry should specify the information as described in the Plug and Play support for HID over SPI article.

Power Management (touchscreen-power-management)

Article • 09/30/2021

This topic provides information about Touchscreen power management, and discusses power consumption requirements.

Power Consumption

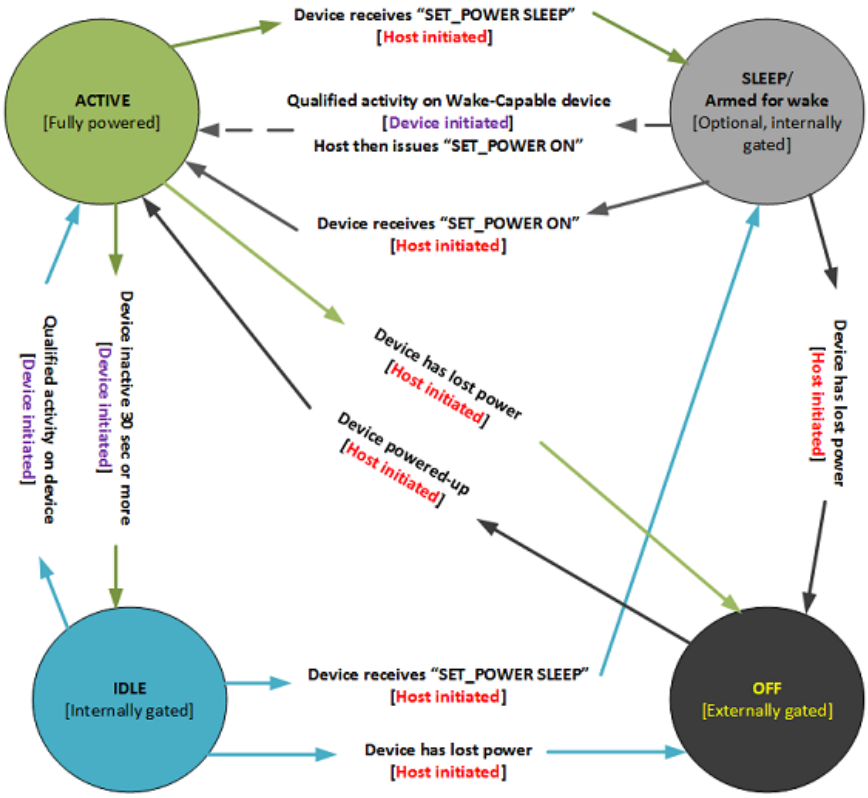
Power consumption requirements for the various power modes of a Windows Touchscreen device are at the discretion of the OEM, or the system builder. However, Windows provides facilities to help balance power efficiency and response latency.

HID I^2C Devices

Windows Touchscreen devices that are connected via I^2C can implement support for up to four distinct power states:

- Active
- Idle
- Either Sleep, or Armed for Wake (both optional)
- Off

The four power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which one or more contacts are present, or there has been activity within the last 30 seconds. When power is applied to a Touchscreen device, then after device boot-up is completed, the device should be ready and in the **Active** power state.

A Touchscreen device should adhere to the *contact down latency* and *contact move latency* requirements for this mode, as this is where the bulk of the compatibility requirements will be tested, and where the majority of user interactions will occur.

Idle State

The **Idle** state is defined as the device operating mode in which no activity has occurred within the last 30 seconds.

A device can elect to reduce its scan rate in this mode, to reduce power consumption while still adhering to the contact down latency requirement for this mode. Once the device has detected activity, it should transition back to the **Active** state.

Sleep (Armed for Wake) State

The **Sleep** state is defined as the operating mode that the device enters, after the host sends a HID I^2C "SET_POWER SLEEP" command to the device.

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of asserting an interrupt to wake the system, in response to a qualified activity. A Windows Touchscreen device should ensure that interrupts are not asserted for spurious contacts, since such contacts would result in an unintended system wake. There are no contact down latency requirements for this mode.

The device should transition to the **Active** state after it receives a HID I^2C "SET_POWER ON" command from the host.

Off State

The **Off** state is defined as the operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed, the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

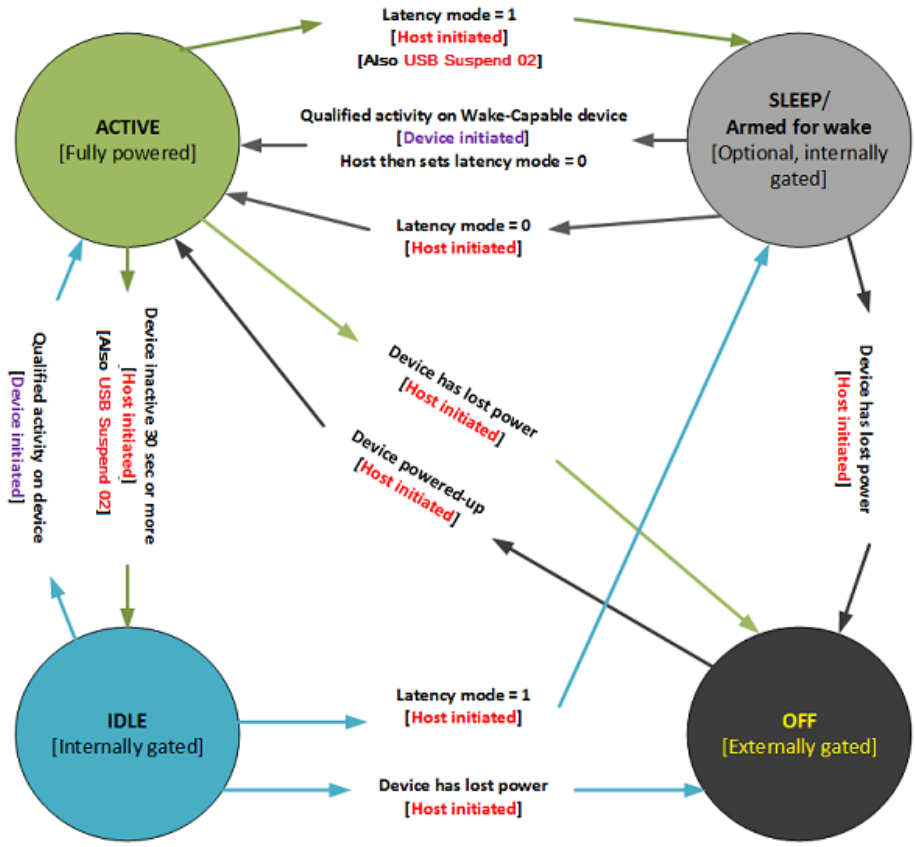
In the **Off** state a device should not consume any power.

USB Devices

Windows Touchscreen devices that connect to the host via USB, can implement support for up to four distinct power states:

- Active
- Idle
- Either Sleep, or Armed for Wake (both optional)
- Off

The four power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which the host has not suspended the device. When power is applied to a Touchscreen device, then after device boot-up is completed, the device should be ready, and in the **Active** power state.

A Touchscreen device should adhere to the contact down latency, and contact move latency requirements for this mode, as this is where the bulk of the compatibility requirements will be tested, and where the majority of user interactions will occur.

Idle State

The **Idle** state is defined as the device operating mode in which no contact activity has occurred within 30 seconds, and the device has therefore been suspended. This is referred to as "*USB selective suspend*."

All Windows Touchscreen devices that are connected to their host via USB, should support selective suspend. Such devices should report this capability via a Microsoft OS descriptor.

A Touchscreen device can elect to reduce its scan rate in this mode, to reduce power consumption, while still adhering to the down latency requirement for this mode.

Once the device has detected qualified activity, it should signal a remote wake. Starting from the time of detection of the activity, the device should buffer at least 100ms worth of contact reports, to ensure that little to no input is lost while the USB host controller is resuming.

Sleep (Armed for Wake) State

The **Sleep** state is defined as the device operating mode in which the host has transitioned to S3 or *Connected Standby*.

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of signaling a remote wake (in response to qualified activity) to wake the system. An Windows Touchscreen device should ensure that remote wake is not signaled in response to spurious contacts, which would result in an unintended system wake. There are no contact down latency requirements for this mode.

A contact occurrence that causes entry into the sleep state should not be reported to wake the system.

Off State

The **Off** state is defined as the device operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed, the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

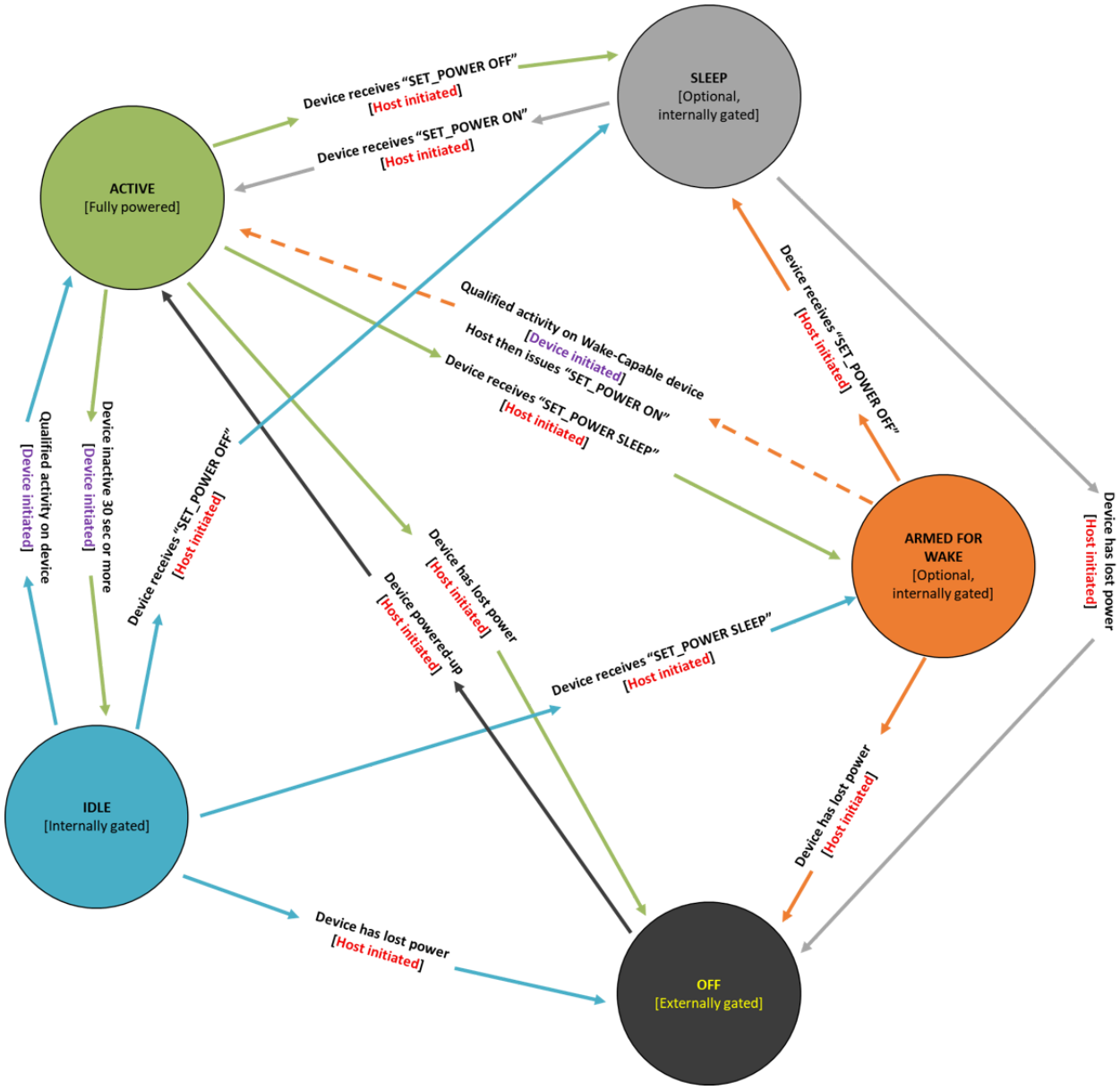
In the **Off** state a Touchscreen device should not consume any power.

SPI Devices

Windows Touchscreen devices that are connected via SPI can implement support for up to five distinct power states:

- Active
- Idle
- Sleep (optional)
- Armed for Wake (optional)
- Off

The five power states are shown in the following diagram, along with the device activities that cause transitions from one state to another.



Active State

The **Active** state is defined as the device operating mode in which one or more contacts are present, or there has been activity within the last 30 seconds. When power is applied to a Touchscreen device, then after device boot-up is completed, the device should be ready and in the **Active** power state.

A Touchscreen device should adhere to the *contact down latency* and *contact move latency* requirements for this mode, as this is where the bulk of the compatibility requirements will be tested, and where the majority of user interactions will occur.

Idle State

The **Idle** state is defined as the device operating mode in which no activity has occurred within the last 30 seconds.

A device can elect to reduce its scan rate in this mode, to reduce power consumption while still adhering to the contact down latency requirement for this mode. Once the

device has detected activity, it should transition back to the **Active** state.

Sleep State

The **Sleep** state is defined as the operating mode that the device enters, after the host sends a HID SPI "SET_POWER OFF" command to the device.

In this state, the device should enter its lowest possible internal power state and should not be scanning for input or be armed to wake.

The device should transition to the **Active** state after it receives a HID SPI "SET_POWER ON" command from the host.

Armed for Wake State

The **Armed for Wake** state is defined as the device operating mode in which the host has transitioned to a lower power state, but is still scanning for input, so it is configured to be woken up by touch interactions.

A device can elect to reduce its scan rate significantly in this mode, to reduce power consumption while still being capable of signaling a remote wake (in response to qualified activity) to wake the system. A Windows Touchscreen device should ensure that remote wake is not signaled in response to spurious contacts, which would result in an unintended system wake. There are no contact down latency requirements for this mode.

A contact occurrence that causes entry into the armed for wake state should not be reported to wake the system.

The device should transition to the **Active** state after it receives a HID SPI "SET_POWER ON" command from the host.

Off State

The **Off** state is defined as the operating mode in which the device has had its power completely removed. When power is applied to the device, then after device boot-up is completed, the device should be ready and in the **Active** power state. Device boot-up should complete before the computer display is initialized and enabled.

In the **Off** state a device should not consume any power.

Protocol Implementation (touchscreen-protocol-implementation)

Article • 03/18/2021

This section provides guidelines for Touchscreen protocol implementation. Windows Touchscreen devices are expected to use the Human Interface Device (HID) protocol to communicate with the host.

A good understanding of the HID protocol is needed to be able to understand the information presented here. See the following resources for information about the HID protocol:

- Device Class Definition for Human Interface Devices (HID) Version 1.11
- HID Usage Tables Version 1.12
- HID Over I2C Protocol Specification Version 1.0

Windows includes a HID class driver and corresponding HID I^2C, HID USB and HID BTH miniport drivers. So there is no need to develop 3rd-party drivers for Windows Touchscreen devices that use one of these buses (unless you are developing or using a 3rd-party bus controller). Just be aware that using a bus that is different from the ones listed, might result in the need for a third party miniport driver to expose the Touchscreen as a HID device to Windows.

You only need to report the usages that are described in this section, in the firmware for your Windows Touchscreen device. Windows will use your firmware and its own HID drivers to enable touch and gesture capabilities for your device, and give Windows applications access to your device.

It is important to note that only touch solutions that use Microsoft inbox drivers are considered compatible. Any solutions that use 3rd-party drivers (either miniport or filter drivers) cannot apply for compatibility.

A sample descriptor is provided in Sample Report Descriptors.

In this section

Topic	Description

Topic	Description
Required HID Descriptors	This topic presents the required HID descriptors (and device attributes) for a Windows Touchscreen device in Windows 10 and later operating systems.
Required HID Top-Level Collections	This topic discusses the required HID top-level collections that are used for Touchscreen reporting in Windows 10 and later operating systems.
Packet Reporting Modes	This topic provides information about the packet reporting modes for a Windows Touchscreen device in Windows 10.
Sample Report Descriptors	This topic presents a sample report descriptor and a sample descriptor excerpt for a Windows Touchscreen device.

Required HID Descriptors (touchscreen-required-hid-descriptors)

Article • 03/18/2021

This topic presents the required HID descriptors (and device attributes) for a Windows Touchscreen device in Windows 10 and later operating systems.

The following table shows the required USB HID descriptor. For more information, see **section 6.2.1** in Device Class Definition for Human Interface Devices (HID) Version 1.11.

Member	Size in bytes	Description
bLength	1	Size of the descriptor
bDescriptorType	1	Type of descriptor
bcdHID	2	HID version number
bCountryCode	1	Country code
bNumDescriptors	1	Number of descriptors
bDescriptorType	1	Descriptor type
bDescriptorLength	2	Length of the descriptor

Required I²C HID Descriptor

The following table shows the required I²C HID descriptor.

Member	Size in bytes	Description
wHIDDescLength	2	The length of the complete HID descriptor (in Bytes).
bcdVersion	2	The version number, in binary coded decimal (BCD) format.
wReportDescLength	2	The length of the Report descriptor (in Bytes).
wReportDescRegister	2	The register index containing the Report descriptor.
wInputRegister	2	The register number to read the input report (in unsigned Bytes).

Member	Size in bytes	Description
wMaxInputLength	2	The length of the largest input report to be read from the input register.
wOutputRegister	2	The register number to send the output (in unsigned Bytes).
wMaxOutputLength	2	The length of the largest output report to be sent.
wCommandRegister	2	The register number to send command requests (in unsigned Bytes).
wDataRegister	2	The register number to exchange data with command requests (in unsigned Bytes).
wVendorID	2	USB-IF assigned Vendor ID.
wDeviceID	2	Device ID.
wVersionID	2	Firmware version number.

Required Device Attributes

The following HID properties must be provided in the device attributes. The reporting of these device attributes is bus-specific. Consult the HID-specific guidance for your choice of bus.

Member	Description	USB	I^2C
wVendorID	Vendor ID	idVendor in USB Device Descriptor	wVendorID in I^2C HID Descriptor (see the preceding table).
wProduct	Product ID	idProduct in USB Device Descriptor	wDeviceID in I^2C HID Descriptor (see the preceding table).
wVersionID	Firmware version number	bcdDevice in USB Device Descriptor	wVersionID I^2C HID Descriptor (see the preceding table).

Required HID Top-Level Collections (touchscreen-required-hid-top-level-collections)

Article • 02/03/2022

This topic discusses the required HID top-level collections that are used for Touchscreen reporting in Windows 10 and later operating systems.

A Windows Touchscreen device should expose, at a minimum, the single mandatory top-level collection for touch reporting. An optional (but recommended) collection for firmware updates can also be implemented.

The following diagram shows the HID collections for a Windows Touchscreen device.

Windows Touchscreen Collection

In Windows 10 a Touchscreen device can use the HID protocol to provide a top-level collection that appears as a digitizer/touchscreen (Page **0x0D**, Usage **0x04**).

The Windows Touchscreen collection provides multi-contact reporting to the host, as well as device information that pertains to those reports. The collection should support one mandatory input report with contact information, and one feature report with the maximum number of contacts that the digitizer supports. An optional (but highly recommended) feature report can be implemented to obtain latency mode hints from the host, to achieve reduced power consumption on USB devices in sleep mode. Lastly, an optional (but recommended) feature report can be implemented to communicate certification status.

Windows Touchscreen Input Reports

Contact Level Usages includes all mandatory usages and supported optional usages that pertain to each unique digitizer contact that is reported. The host uses the following usages (via the Windows Touchscreen collection) to extract contact data from an input report.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Contact ID	Uniquely identifies the contact within a given frame.	0x0D	0x51	Mandatory
X	X coordinate of contact position.	0x01	0x30	Mandatory for T Optional for C
Y	Y coordinate of contact position.	0x01	0x31	Mandatory for T Optional for C
Tip	Set, if the contact is on the surface of the digitizer.	0x0D	0x42	Mandatory
Confidence	Set, when a contact is too large to be a finger.	0x0D	0x47	Optional
Width	Width of bounding box around a contact.	0x0D	0x48	Optional

Member	Description	Page	ID	Mandatory/Optional
Height	Height of bounding box around a contact.	0x0D	0x49	Optional
Pressure	Amount of pressure the user is applying to the contact point.	0x0D	0x30	Optional
Azimuth	The counter-clockwise rotation of the cursor about the Z-axis.	0x0D	0x3F	Optional

The following table shows all the mandatory report level usages that should be present in all input reports for a Windows Touchscreen device.

Expand table

Member	Description	Page	ID	Mandatory/Optional
Report ID	Windows Touchscreen Report ID.	0x0D	0x05	Mandatory
Scan Time	Relative scan time per frame.	0x0D	0x56	Optional
Contact Count	Total number of contacts that are reported in a given report.	0x0D	0x54	Mandatory

Any device that does not report all mandatory usages at either the contact or report level, will be non-functional as a Windows Touchscreen device. Mandatory usages are strictly enforced by the Windows host. Where a logical maximum value has not been restricted, it can be optimized to reduce descriptor size.

Contact ID

Contact ID uniquely identifies a contact in a report for its lifecycle. The contact ID must remain constant while the contact is detected and reported by the device. Each separate concurrent contact must have a unique identifier. Identifiers can be reused after the previously associated contact is no longer detected or reported. There is no expected numeric range and the values that are used are only limited by the specified logical maximum in the descriptor.

X/Y

X and Y report the coordinates of a given contact. A device can report two points for each contact. The first point (known as T) is considered to be the point that the user intended to touch, while the second point (known as C) is considered to be the center of the contact. Devices that are capable of reporting T and C should have a usage array of two X values and two Y values. The values in the first position in the arrays are interpreted as the coordinates for T and the values in the second position are interpreted as the coordinates for C. The report count for both usages is 2, to indicate the presence of a usage array.

Devices that report C must also report the Width and Height usages. The host uses C to build the bounding rectangle around the contact. If the device only reports one X and Y pair, the host uses that pair for T and C. The sample touch descriptor includes usage arrays for both X and Y.

A device that reports only T, must not have a usage array for the X and Y properties. In other words, the report count for each usage is 1 as indicated in the following extracts from the sample descriptor. The excerpts also illustrate the difference between a device that supports only T, and a device that supports T and C.

```
    0x05, 0x01,                    //      USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,              //      LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                    //      REPORT_SIZE (16)
    0x55, 0x0e,                    //      UNIT_EXPONENT (-2)
    0x65, 0x13,                    //      UNIT(Inch,EngLinear)
    0x09, 0x30,                    //      USAGE (X)
    0x35, 0x00,                    //      PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,              //      PHYSICAL_MAXIMUM (1205)
    0x95, 0x01,                    //      REPORT_COUNT (1)
    0x81, 0x02,                    //      INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,              //      PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                    //      USAGE (Y)
    0x81, 0x02,                    //      INPUT (Data,Var,Abs)
```

A device that supports T and C, uses usage arrays for reporting the X and Y values. The report count for both X and Y is 2.

```
    0x05, 0x01,                    //      USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,              //      LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                    //      REPORT_SIZE (16)
    0x55, 0x0e,                    //      UNIT_EXPONENT (-2)
    0x65, 0x13,                    //      UNIT(Inch,EngLinear)
```

```
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,              //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                    //       REPORT_COUNT (2)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,              //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                    //       USAGE (Y)
    0x81, 0x02,
```

Note These examples take advantage of the HID rule that global items stay the same for every main item, until they are changed. This allows both X and Y usages to share just one entry for the report count.

The following global items must be specified for the X and Y usages:

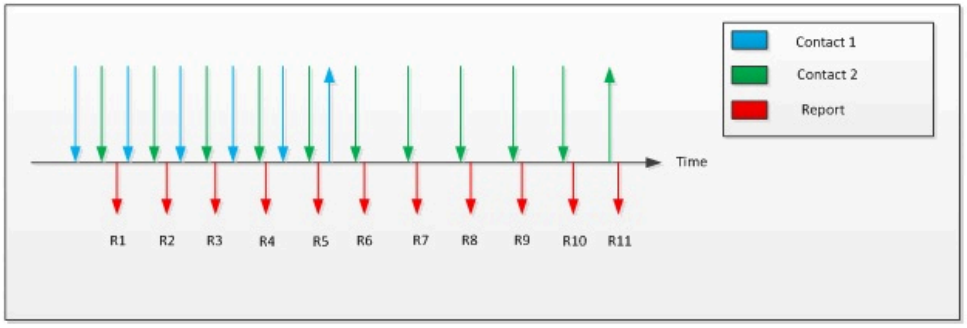
- Logical minimum
- Logical maximum
- Physical minimum
- Physical maximum
- Unit
- Unit Exponent

The physical range for the device and the units must be accurately reported. If the information is inaccurate, the device will not work correctly. Devices must also report data within the logical range that is specified in the report descriptor. Any reported value outside this range will be considered as invalid data and the value will be changed to the nearest boundary value (logical minimum or logical maximum).

Tip

The tip switch is used to indicate when the contact is on the surface or has left the surface of the digitizer. This is indicated by a main item with a report size of 1 bit. When delivering a contact report, the bit should be set when the contact is on the digitizer surface, and cleared when the contact has left the surface.

When a contact is being reported with the tip switch clear, the X/Y location that is reported should be the same as the last position that was reported with the tip switch set.



As shown in the preceding diagram, two contacts are placed on a Windows Touchscreen. Sometime later, the first contact is lifted while the second remains on the surface. This would be reported as described in the following table.

Expand table

Report	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Contact count	2	2	2	2	2	2	1	1	1	1	1
Contact 1: tip switch	1	1	1	1	1	0	NR	NR	NR	NR	NR
Contact 1: X, Y	X_1, Y_1	X_2, Y_2	X_3, Y_3	X_4, Y_4	X_5, Y_5	X_6, Y_6	NR	NR	NR	NR	NR
Contact 2: tip switch	1	1	1	1	1	1	1	1	1	1	0
Contact 2: X, Y	X_1, Y_1	X_2, Y_2	X_3, Y_3	X_4, Y_4	X_5, Y_5	X_6, Y_6	X_7, Y_7	X_8, Y_8	X_9, Y_9	X_{10}, Y_{10}	X_{11}, Y_{11}

The preceding table shows the report sequence for two contacts with separated lift (Two-Finger Hybrid).

Confidence

Confidence is a suggestion from the device about whether the touch contact was an intended, or an accidental touch. If you are confident that the touch is intended, then set the confidence usage to 1 (true) for the duration of the interaction, including lift (when the tip switch is cleared). Your device should reject accidental touches as thoroughly as it can while the latency stays within the required range. If you are not certain that the touch is intended, and your device did not reject the touch as accidental, then change the confidence usage to 0 (false), allowing the operating system to cancel

the touch contact as needed. If your device always rejects accidental touches, you do not need to include the confidence usage.

Width and Height

The Width and Height usages represent the width and height of the bounding box around the touch contact. The reported values should never be zero (0) except when an "UP" event is being reported, in which case they should be zero (0).

Pressure

Pressure is a measurement of the force that the finger exerts against the digitizer surface. There are no restrictions on the range allowed for pressure.

Azimuth

Azimuth specifies the counter-clockwise rotation of the cursor around the Z-axis through a full circular range. The physical range and logical range must be specified. The physical range must be 0 to 360 or equivalent in radians, however the device may report either the full range or half the range assuming the touch ellipsis is symmetric based on hardware capabilities. The logical range must be large enough to deliver data that is accurate to at least two decimal places.

Scan time

Scan Time reports relative digitizer time in 100μs units. It represents the delta from the first frame that was reported after a device starts reporting data subsequent to a period of inactivity. The first scan time received is treated as a base time for subsequent reported times. The deltas between reported scan times should reflect the scanning frequency of the digitizer. It is important to note that unlike other usages, the host does not allow any flexibility for the unit for the scan time usage if it is implemented. It must be in 100μs units. The value is expected to roll over, as only 2 bytes are allocated to the counter.

The scan time value should be the same for all contacts within a frame.

Contact count

Contact count is used to indicate the number of contacts that are being reported in a given frame, regardless of the tip switches with which the contacts are associated.

Windows Touchscreen Feature Reports

The host uses the following usages (via the Windows Touchscreen collection) to extract contact data from an input report.

Contact Count Maximum Feature Report

This report specifies the total number of contacts that a multi-touch device supports. A Windows Touchscreen device should report this value via specification of the contact count maximum (Page **0x0D**, Usage **0x55**) in the contact count maximum feature report. While reporting data, a device must not report more contacts than the contact count maximum. While reporting data, a device must not report more contacts than the Contact Count maximum. If an input frame contains more contacts than the maximum, the host will discard the entire frame, including the data for any previously reported contacts. For the best user experience, the device should suppress the reporting of any new contacts detected after the maximum number has been reached for the lifetime of the new contact, even if any existing contacts depart. Existing contacts can continue to be reported, and if any depart, newly arriving contacts can take their place.

Device Certification Status Feature Report

When a device passes the compatibility requirements, Microsoft will issue a cryptographically signed binary blob (known as a THQA blob) to the device's manufacturer . The manufacturer will place this blob into the device's firmware prior to production. When a touch device attempts to connect, the signature will be verified by the Windows operating system. Windows will only do this for devices that expose the THQA feature report in their descriptor.

Note Reporting a valid certified blob to the host is optional in Windows 10, but is necessary for backwards compatibility with Windows 8 and Windows 8.1. A device with a THQA blob will include the phrase "Full Windows Touch Support" in the control panel. A device without a certified THQA blob will indicate the phrase "Windows Touch Support" in the control panel.

The blob itself will consist of 256 bytes of binary data, and should be reported as illustrated by the following HID descriptor. Device manufacturers should be sure that before they issue the signed binary blob from Microsoft, the sample blob provided below is presented to Windows instead.

```
    0x06, 0x00, 0xff,                   //     USAGE_PAGE (Vendor Defined)
    0x09, 0xC5,                         //     USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //     LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //     REPORT_SIZE (8)
    0x96, 0x00, 0x01,                   //     REPORT_COUNT (0x100 (256
```

The following is the sample blob in clear text.

```
0xfc, 0x28, 0xfe, 0x84, 0x40, 0xcb, 0x9a, 0x87, 0x0d, 0xbe, 0x57, 0x3c,
0xb6, 0x70, 0x09, 0x88,
0x07, 0x97, 0x2d, 0x2b, 0xe3, 0x38, 0x34, 0xb6, 0x6c, 0xed, 0xb0, 0xf7,
0xe5, 0x9c, 0xf6, 0xc2,
0x2e, 0x84, 0x1b, 0xe8, 0xb4, 0x51, 0x78, 0x43, 0x1f, 0x28, 0x4b, 0x7c,
0x2d, 0x53, 0xaf, 0xfc,
0x47, 0x70, 0x1b, 0x59, 0x6f, 0x74, 0x43, 0xc4, 0xf3, 0x47, 0x18, 0x53,
0x1a, 0xa2, 0xa1, 0x71,
0xc7, 0x95, 0x0e, 0x31, 0x55, 0x21, 0xd3, 0xb5, 0x1e, 0xe9, 0x0c, 0xba,
0xec, 0xb8, 0x89, 0x19,
0x3e, 0xb3, 0xaf, 0x75, 0x81, 0x9d, 0x53, 0xb9, 0x41, 0x57, 0xf4, 0x6d,
0x39, 0x25, 0x29, 0x7c,
0x87, 0xd9, 0xb4, 0x98, 0x45, 0x7d, 0xa7, 0x26, 0x9c, 0x65, 0x3b, 0x85,
0x68, 0x89, 0xd7, 0x3b,
0xbd, 0xff, 0x14, 0x67, 0xf2, 0x2b, 0xf0, 0x2a, 0x41, 0x54, 0xf0, 0xfd,
0x2c, 0x66, 0x7c, 0xf8,
0xc0, 0x8f, 0x33, 0x13, 0x03, 0xf1, 0xd3, 0xc1, 0x0b, 0x89, 0xd9, 0x1b,
0x62, 0xcd, 0x51, 0xb7,
0x80, 0xb8, 0xaf, 0x3a, 0x10, 0xc1, 0x8a, 0x5b, 0xe8, 0x8a, 0x56, 0xf0,
0x8c, 0xaa, 0xfa, 0x35,
0xe9, 0x42, 0xc4, 0xd8, 0x55, 0xc3, 0x38, 0xcc, 0x2b, 0x53, 0x5c, 0x69,
0x52, 0xd5, 0xc8, 0x73,
0x02, 0x38, 0x7c, 0x73, 0xb6, 0x41, 0xe7, 0xff, 0x05, 0xd8, 0x2b, 0x79,
0x9a, 0xe2, 0x34, 0x60,
0x8f, 0xa3, 0x32, 0x1f, 0x09, 0x78, 0x62, 0xbc, 0x80, 0xe3, 0x0f, 0xbd,
0x65, 0x20, 0x08, 0x13,
0xc1, 0xe2, 0xee, 0x53, 0x2d, 0x86, 0x7e, 0xa7, 0x5a, 0xc5, 0xd3, 0x7d,
0x98, 0xbe, 0x31, 0x48,
0x1f, 0xfb, 0xda, 0xaf, 0xa2, 0xa8, 0x6a, 0x89, 0xd6, 0xbf, 0xf2, 0xd3,
0x32, 0x2a, 0x9a, 0xe4,
0xcf, 0x17, 0xb7, 0xb8, 0xf4, 0xe1, 0x33, 0x08, 0x24, 0x8b, 0xc4, 0x43,
0xa5, 0xe5, 0x24, 0xc2
```

Latency Mode Feature Report

Note Windows 10, version 1511 and earlier operating systems do not currently issue this HID feature report to Pen or Touchscreen devices that happen to provide support for it. However, adding support for the *Latency Mode Feature Report* to a device, will make sure that the device is ready for the future date when Windows supports it on Pen and Touchscreen devices.

The latency mode feature report is sent by the host to a Windows Touchscreen device, to indicate when high latency is desirable for power savings, and conversely, when normal latency is desired for operation. For USB-connected Windows Touchscreen devices, this allows the device to differentiate between being suspended for inactivity (runtime IDLE), and being suspended because the system is entering S3 or Connected Standby.

The latency mode should be indicated via the value for the latency mode usage (Page **0x0D**, Usage **0x60**) in the latency mode feature report.

Expand table

Latency mode value	Latency mode
0	Normal latency
1	High latency

Firmware Update Collection (Optional)

A Windows Touchscreen device can use the HID protocol to provide a vendor-specific top-level collection for performing device firmware, and vendor configuration updates.

The vendor-specific firmware update collection can provide an output report for transferring the firmware payload from the host to the device.

This is highly advantageous as it allows for firmware updates to be performed without requiring a driver on the host. It is mandatory for the *wVersionID* device attribute to be incremented after a firmware upgrade.

A Windows Touchscreen device should be able to recover from a firmware update that failed due to power loss (or some other error), if you shut down, and then reapply its power supply. It is highly recommended that basic functionality be available even after a failed firmware update.

Packet Reporting Modes

Article • 01/18/2019

This topic provides information about the packet reporting modes for a Windows Touchscreen device in Windows 10.

Parallel mode

In Parallel mode, devices report all contact information in a single packet. Each physical contact is represented by a logical collection that is embedded in the top-level collection. This logical collection contains all the usages that the device supports for each contact. Each of the logical collections must be identical when you use Parallel mode. Because the device generally reports fewer contacts than the maximum, the number of contacts that are reported in a parallel packet should be communicated by using the Contact Count usage.

For example, consider a device that supports three contacts. If the user has only two fingers on the digitizer, then the parallel packet will consist of information about only those two contacts, in a report that can carry data for three contacts. In this case, the Contact Count should be set to two, so that the client application knows that any information about more than two contacts is not valid.

A disadvantage of reporting multiple contacts in one report is that space is wasted per report every time that there are fewer contacts than the maximum number of contacts possible. Devices can use the Hybrid mode to reduce this inefficiency.

Hybrid mode

In Hybrid mode, the number of contacts that can be reported in one report is less than the maximum number of contacts that the device supports. For example, a device that supports a maximum of 4 concurrent physical contacts, can set up its top-level collection to deliver a maximum of two contacts in one report. If four contact points are present, the device can break these up into two serial reports that deliver two contacts each.

When a device delivers data in this manner, the Contact Count usage value in the first report should reflect the total number of contacts that are being delivered in the hybrid reports. The other serial reports should have a contact count of zero (0).

Single finger hybrid reporting mode

The first input report for a given frame should indicate the total number of contacts that are to be reported by using the Contact Count usage. And all subsequent input reports for the same frame should have a value of zero (0) for the Contact Count usage, to indicate that they are part of the previously reported frame. The scan time for all reports of a given frame should be identical.

Here's a table showing the reporting sequence for a two-contact scenario, and using a single finger hybrid reporting mode.

Report	**1**	**2**	**3**	**4**	**5**	**6**
Contact Count	2	0	2	0	2	0
Contact ID	1	2	1	2	1	2
Contact 1: X, Y	X_1, Y_1	NR	X_2, Y_2	NR	X_3, Y_3	NR
Contact 1: Tip switch	1	NR	1	NR	1	NR
Contact 2: X, Y	NR	X_1, Y_1	NR	X_2, Y_2	NR	X_3, Y_3
Contact 2: Tip switch	NR	1	NR	1	NR	1
Scan Time	T1	T1	T2	T2	T3	T3

Sample Report Descriptors (touchscreen-sample-report-descriptors)

Article • 12/05/2022

This topic presents a sample report descriptor and a sample descriptor excerpt for a Windows Touchscreen device.

Note If you are implementing the following descriptor on a SKU that is specific to a mobile device, or a Windows phone, then see the knowledge base article about the HID descriptor issue (partner access required), to make sure that you have the right setup on your mobile device.

> **Note**
>
> Although vendor specific usages may be used at any level of the report, they must be used in a way that could be implemented equally across all reporting methods, namely, serial, parallel, and hybrid. Not adhering to this guidance may not result in an HLK failure but could result in serious compatibility issues.

Sample Report Descriptor

The following is a sample descriptor of a two-finger parallel/hybrid mode device.

```
    0x05, 0x0d,                    // USAGE_PAGE (Digitizers)
    0x09, 0x04,                    // USAGE (Touch Screen)
    0xa1, 0x01,                    // COLLECTION (Application)
    0x85, 0x01,                    //   REPORT_ID (Touch)
    0x09, 0x22,                    //   USAGE (Finger)
    0xa1, 0x02,                    //     COLLECTION (Logical)
    0x09, 0x42,                    //       USAGE (Tip Switch)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //       REPORT_SIZE (1)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x07,                    //       REPORT_COUNT (7)
    0x81, 0x03,                    //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                    //       REPORT_SIZE (8)
    0x09, 0x51,                    //       USAGE (Contact Identifier)
```

```
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x01,                    //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,              //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                    //       REPORT_SIZE (16)
    0x55, 0x0e,                    //       UNIT_EXPONENT (-2)
    0x65, 0x13,                    //       UNIT(Inch,EngLinear)
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,              //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                    //       REPORT_COUNT (2)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,              //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                    //       USAGE (Y)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x0d,                    //       USAGE_PAGE (Digitizers)
    0x09, 0x48,                    //       USAGE (Width)
    0x09, 0x49,                    //       USAGE (Height)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x55, 0x0C,                    //       UNIT_EXPONENT (-4)
    0x65, 0x12,                    //       UNIT (Radians,SIROtation)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x47, 0x6f, 0xf5, 0x00, 0x00,  //       PHYSICAL_MAXIMUM (62831)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x27, 0x6f, 0xf5, 0x00, 0x00,  //       LOGICAL_MAXIMUM (62831)
    0x09, 0x3f,                    //       USAGE
(Azimuth[Orientation])
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0xc0,                          //     END_COLLECTION
    0x09, 0x22,                    //   USAGE (Finger)
    0xa1, 0x02,                    //     COLLECTION (Logical)
    0x09, 0x42,                    //       USAGE (Tip Switch)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //       REPORT_SIZE (1)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x07,                    //       REPORT_COUNT (7)
    0x81, 0x03,                    //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                    //       REPORT_SIZE (8)
    0x09, 0x51,                    //       USAGE (Contact Identifier)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x01,                    //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,              //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                    //       REPORT_SIZE (16)
    0x55, 0x0e,                    //       UNIT_EXPONENT (-2)
    0x65, 0x13,                    //       UNIT(Inch,EngLinear)
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,              //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                    //       REPORT_COUNT (2)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,              //       PHYSICAL_MAXIMUM (906)
```

```
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x05, 0x0d,                         //       USAGE_PAGE (Digitizers)
    0x09, 0x48,                         //       USAGE (Width)
    0x09, 0x49,                         //       USAGE (Height)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x55, 0x0C,                         //       UNIT_EXPONENT (-4)
    0x65, 0x12,                         //       UNIT (Radians,SIROtation)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x47, 0x6f, 0xf5, 0x00, 0x00,       //       PHYSICAL_MAXIMUM (62831)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x27, 0x6f, 0xf5, 0x00, 0x00,       //       LOGICAL_MAXIMUM (62831)
    0x09, 0x3f,                         //       USAGE
(Azimuth[Orientation])
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0xc0,                               //     END_COLLECTION
    0x05, 0x0d,                         //   USAGE_PAGE (Digitizers)
    0x55, 0x0C,                         //     UNIT_EXPONENT (-4)
    0x66, 0x01, 0x10,                   //     UNIT (Seconds)
    0x47, 0xff, 0xff, 0x00, 0x00,       //       PHYSICAL_MAXIMUM (65535)
    0x27, 0xff, 0xff, 0x00, 0x00,       //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                         //   REPORT_SIZE (16)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x09, 0x56,                         //   USAGE (Scan Time)
    0x81, 0x02,                         //   INPUT (Data,Var,Abs)
    0x09, 0x54,                         //   USAGE (Contact count)
    0x25, 0x7f,                         //   LOGICAL_MAXIMUM (127)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x75, 0x08,                         //   REPORT_SIZE (8)
    0x81, 0x02,                         //   INPUT (Data,Var,Abs)
    0x85, REPORTID_MAX_COUNT,           //   REPORT_ID (Feature)
    0x09, 0x55,                         //   USAGE(Contact Count Maximum)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x25, 0x02,                         //   LOGICAL_MAXIMUM (2)
    0xb1, 0x02,                         //   FEATURE (Data,Var,Abs)
    0x85, 0x44,                         //   REPORT_ID (Feature)
    0x06, 0x00, 0xff,                   //   USAGE_PAGE (Vendor Defined)
    0x09, 0xC5,                         //   USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //   LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //   REPORT_SIZE (8)
    0x96, 0x00,  0x01,                  //   REPORT_COUNT (0x100 (256))
    0xb1, 0x02,                         //   FEATURE (Data,Var,Abs)
    0xc0,                               // END_COLLECTION
```

Note that Windows expects the coordinate (0,0) to be the top-left corner of the native display orientation – either portrait or landscape.

Sample Report Descriptor Excerpt - (Latency Mode Feature Report)

The following excerpt is from the descriptor for the Windows Touchscreen top-level collection for the optional (highly recommended) support of the latency mode feature report.

```
    0x05, 0x0d,                         //    USAGE_PAGE (Digitizer)
    0x85, REPORTID_LATENCY,             //    REPORT_ID (Latency)
    0x09, 0x60,                         //    USAGE(Latency Mode)
    0x75, 0x01,                         //    REPORT_SIZE (1)
    0x95, 0x01,                         //    REPORT_COUNT (1)
    0x15, 0x00,                         //    LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //    LOGICAL_MAXIMUM (1)
    0xb1, 0x02,                         //    FEATURE (Data,Var,Abs)
    0x95, 0x07,                         //    REPORT_COUNT (7)
    0xb1, 0x03,                         //    FEATURE (Cnst,Var,Abs)
```

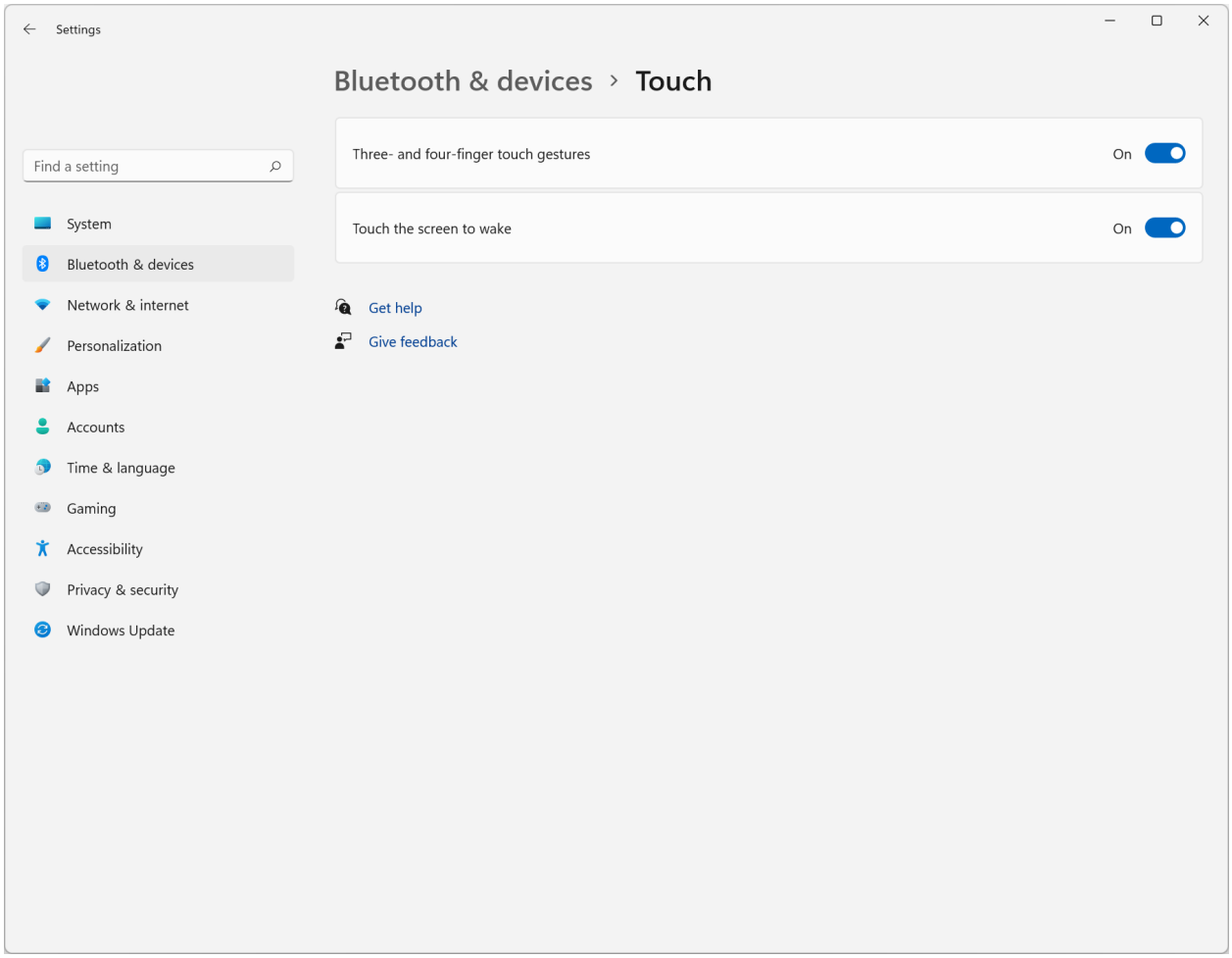
Wake-on-Touch Implementation Guide

Article • 09/28/2022

Introduction

Wake-on-Touch is a new, *optional* capability coming to Windows 11 devices. This feature allows users to touch a device's screen with their finger to wake it from sleep. Using a pen to wake the screen is not supported. Wake-on-Touch is only available on devices that explicitly opt in per the guidance below.

If a device has opted in to Wake-on-Touch, the OEM may specify the default state (enabled/disabled—additional information is in the **OEM Considerations** section below). Users also have a "Touch the screen to wake" option to toggle it on or off based on their preference in the new **Bluetooth & devices -> Touch** settings page per the figure below.



> **Note**
>
> If this functionality is not supported on a device, the setting will not appear on this page.

This document describes how HID based touch devices compatible with Windows can support Wake-On-Touch and what OEMs/ODMs and touch IHVs need to do depending on whether they are using an inbox HID mini-port driver available for USB, I2C or SPI or a 3P HID mini-port specifically for a bus which lacks an inbox driver (i.e. Intel THC SPI).

OEM Considerations

It is imperative that OEMs consider whether to implement Wake-on-Touch for each design and SKU as there are noticeable power consumption trade-offs for the feature. Pressing a key on the keyboard, interacting with the touchpad, or clicking the Bluetooth button on a paired pen, are all methods to wake the device with lower power consumption.

Wake Gesture

When enabling Wake-on-Touch, OEMs can specify the wake gesture to be either a single-tap or a double-tap working with their touch IHV. Whichever gesture is chosen shall be usable anywhere on the active area of the display to wake the device. Once the touch controller has detected a wake based on either the single tap or double tap gesture:

1. Touch controller is required to send 2 frames representative of a single finger down and up (X, Y, TIP SET, followed by X, Y, TIP CLEAR) regardless of what gesture was used to cause the wake
2. The host will only use these 2 frames as indication of user presence to turn on displays, etc. and will not deliver that input to shell or applications
3. The up and down frames can both be sent immediately, however if scantime usage is present it should be incremented between frames

> ⓘ **Note**
>
> If the touch controller has detected a wake gesture and asserts an interrupt to indicate to the host that input report(s) are ready, but the host does not in fact issue the read, the touch controller should revert back to low power state after some vendor defined timeout. This is considered an abnormal case, but may be desirable for touch IHV to account for in order to minimize unnecessary power consumption in standby.

Device Posture

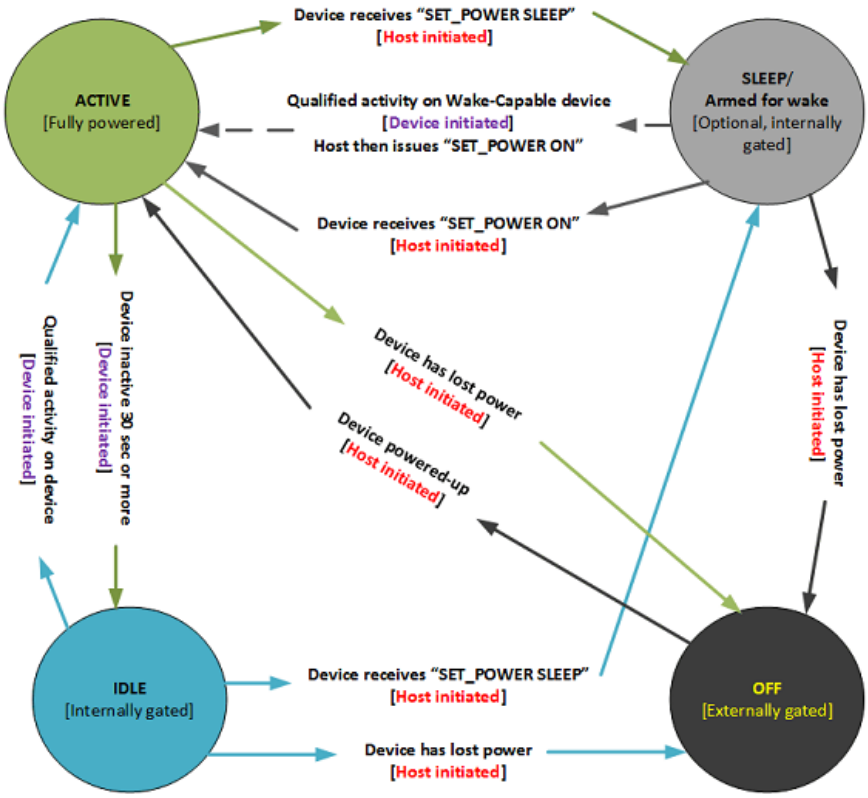
Different postures, device types, and power states should all be taken into consideration for maximum optimization of battery life and reliable UX if the feature is to be utilized. As an example, a convertible device may only want to enable Wake-on-Touch when it is in its “tablet” state and disable it when it is in its “laptop” state in order to conserve battery.

Availability of Wake-on-Touch, be it whenever the device is in standby or just in specific postures requires different approaches in implementation.

For systems with a lid (e.g. a laptop form factor device), the HID stack will automatically disable Wake-on-Touch when the lid is closed. If your device matches this form factor or if it is desired for Wake-on-Touch to be enabled regardless of device posture, **skip the rest of the content in this section and follow the guidance in the rest of this document**.

If it is desired for Wake-on-Touch to only be enabled in certain postures, the recommendation is to implement a filter driver that controls whether the touch controller is armed for wake based on the device posture. In this type of implementation, the filter driver would decide whether to let the IRP flow to ACPI to put the device to D2 (armed for wake) or whether to have the device be put to D3 Cold (off).

If the device design does not permit ACPI to pull power (D3 Cold) from the touch controller when Wake-on-Touch is not desired based on posture then a vendor-specific mechanism should be implemented to allow the touch controller to remain powered (D3 Hot) while informing it to go to its lowest power state, internally gating, and not scanning for any touches. This vendor specific mechanism allows the touch controller to disambiguate whether to be scanning for user input or not in the SLEEP/Armed for wake state per the figure below.



> ⓘ **Note**
>
> This diagram is for HIDI2C. For more information about power transitions, including for HIDUSB and HIDSPI, please see **Touchscreen Power Management**.

Lastly, if it is desired for a device to have different Wake-on-Touch behaviors based on posture, it is recommended to wake the device when a posture change happens. Posture changes are a clear indication of user input and this also ensures the proper arming/disarming of the touch controller when the device enters a new posture.

Default State for Wake-On-Touch User Setting

The default for the Wake-On-Touch setting is ON for capable systems, but an OEM can add a registry value specifying the default should instead be OFF. The change should be applied to the default user and it will get picked up by each new user on the device. Once the user changes the setting, their preference always takes precedence regardless of what the initial OEM default was.

To do so, create the below registry key and store it in the Default User NTUSER.DAT:

Registry Location:
\HKEY_CURRENT_USER\Software\Microsoft\Input\WakeableInputTypes

Registry Key:

Name	Type	Value
Touch	REG_DWORD	0 = Disabled, 1 = Enabled

Requirements for Enabling Wake-on-Touch

General Requirements

It is generally recommended that the device indicates support for wake from D2 state, so that when the operating system (OS) needs to arm the device for wake on touch, it can place it into D2. If the OS does not need to arm the device for wake on touch, it will place the device into D3. This will then allow the device to transition into D3Cold for greater power savings. This can be done by following the _S0W related guidance provided in the ACPI section below. Device-initiated power optimizations (that are not initiated by the host or operating system) should be done in a manner transparent to the operating system.

Please note when the operating system (or host) places the device into a state that enables “Wake-On-Touch” as described in this documentation, it is up to the touch controller to configure itself to wake on an appropriate touch gesture, based on the OEM requirements and/or form-factor of the device. For example, a device with a touch screen that is not occluded by a lid may need to ensure it does not cause spurious wakes.

If a HID touch device has any other top-level HID collections, such as vendor-defined collections, to which OEM software may open file handles to perform device I/Os, closing the file handles as soon as the OEM software completes these device I/Os can reduce power consumption of the HID device. Please note that Windows 11 22H2 has been updated to manage the power state of a Wake-on-Touch device more efficiently by powering it off without allowing wake from modern standby as long as Wake-on-Touch is not needed, for example, when the “Touch the screen to wake” setting is Off or when the laptop lid is closed. For such OEM software, the side effect is that inputs from such top-level HID collections will be suppressed too during modern standby.

ACPI Firmware Requirements

If the touch controller device is defined in ACPI, it must implement the following to report its wake capabilities and resources.

1. _S0W method, which returns the lowest device power state under which the device may signal wake.
2. _CRS method, which defines the wake-capable interrupt.
3. _PRx methods, which define the power resources that are associated with each D-state supported.

For more information, please refer to the Wake-capable interrupts (_CRS) documentation.

Driver Requirements for Enabling Wake-on-Touch

For details on how to implement Wake-on-Touch for either inbox drivers (HIDI2C, HIDSPI, and HIDUSB) or custom drivers, please see the respective documentation below:

Topic	Description
Enabling Wake-on-Touch with Inbox Drivers	This topic discusses how to implement Wake-on-Touch with the inbox HIDI2C, HIDSPI, or HIDUSB drivers.
Enabling Wake-on-Touch with Custom HID Mini Drivers	This topic discusses how to implement Wake-on-Touch with custom HID mini drivers.

Glossary

Term/Abbreviation	Definition
HID	Human Interface Devices.
HIDClass	The Windows inbox class driver for HID.
HID Mini Driver	A Windows HID transport driver that binds to and works with HIDClass. For more information, please refer to the Minidrivers and the HID class driver documentation.
D-State, D0, D2, D3hot, D3Cold	The device power states that are defined by ACPI specification.
Dx	One of the D-states. The actual “x” number is unspecified by this document.
DeviceNotInUse	A condition that occurs when the operating system moves to a state under which it is not consuming (touch) input. An example is laptop lid closure.
DeviceInUse	Opposite to DeviceNotInUse, it is a condition that occurs when the operating system moves to a state under which it may consume (touch) input. An example is that the laptop lid open.

Term/Abbreviation	Definition
Monitor, Screen, and Display	They are sometimes used interchangeably.
MonitorOn	A condition that occurs when the operating system turns on the primary monitor.
MonitorOff	Opposite to MonitorOn. It is a condition that occurs when the operating system turns off the primary monitor.
IRP	I/O Request Packet. For more information, please refer to the I/O request packets documentation.
D-IRP	An IRP that requests to set the device power state.
HIDI2C	The Windows inbox HID mini driver HIDI2C.SYS, which implements I2C transport layer for HID devices according to the HIDI2C specification.
HIDSPI	The Windows inbox HID mini driver HIDSPI.SYS, which implements SPI transport layer for HID devices according to the HIDSPI specification.
HIDUSB	The Windows inbox HID mini driver HIDUSB.SYS, which implements USB transport layer for HID devices according to the HID and USB specifications.
ACPI Firmware	The ACPI codes that are implemented in the system firmware (BIOS).
Extension INF	A new type of Windows INF files in Windows 10. For more information, please refer to the Using an Extension INF File documentation.

Enabling Wake-on-Touch With Inbox Drivers

Article • 10/05/2021

This topic discusses how to implement Wake-on-Touch with the inbox HIDI2C, HIDSPI, or HIDUSB drivers.

Inbox HIDI2C Driver

The inbox HIDI2C driver sends the command SetPower(Sleep) whenever the device enters a lower power state. That happens to both transitions to D2 and D3.

How to Enable Wake-on-Touch

1. Define wake resources and capabilities in ACPI as per the **ACPI Firmware Requirements** section in the Wake-on-Touch Implementation Guide.
2. In response to the SetPower(Sleep) command, the touch controller firmware should enable Wake-on-Touch.
3. In response to the transition from D0 to D3Hot, although a SetPower(Sleep) command is still sent to the device, the ACPI firmware should leave the minimal power resources on the touch controller, since Wake-on-Touch is not needed.
4. If D3Cold is supported, in response to the transition from D3Hot to D3Cold, the ACPI firmware may shut all the power resources of the touch controller.
5. Build an INF to opt-in the Wake-on-Touch policy as per the **INF Requirements** section below.
6. Validate that the device wakes up on the appropriate touch gesture(s), based on the OEM requirements and/or form-factor of the device. Also validate that it does not cause spurious wake events and the power consumption from enabling the feature is acceptable. See Wake-on-Touch Validation Guide.

Inbox HIDSPI Driver

The HIDSPI driver sends SetPower(Sleep) command when the device is entering D2. This command is defined in the HIDSPI specification.

For D3(Hot), the HIDSPI driver sends SetPower(Off) command, instead of SetPower(Sleep). This command is defined in the HIDSPI specification.

How to Enable Wake-on-Touch

1. Define wake resources and capabilities in ACPI as per the **ACPI Firmware Requirements** in the Wake-on-Touch Implementation Guide.
2. In response to the SetPower(Sleep) command, the touch controller firmware should enable Wake-on-Touch.
3. In response to the SetPower(Off) command, the touch controller firmware should enter the Off state where minimal power is consumed, just in case D3Cold is not supported and the touch controller may stay at D3Hot forever.
4. If D3Cold is supported, in response to the transition from D3Hot to D3Cold, the ACPI firmware may shut all the power resources of the touch controller.
5. Build an INF to opt-in the Wake-on-Touch policy as per the **INF Requirements** section below.
6. Validate that the device wakes up on the appropriate touch gesture(s), based on the OEM requirements and/or form-factor of the device. Also validate that it does not cause spurious wake events and the power consumption from enabling the feature is acceptable. See Wake-on-Touch Validation Guide.

Inbox HIDUSB Driver

How to Enable Wake-on-Touch

1. Build an INF to opt-in the Wake-on-Touch policy as per the **INF Requirements** section below.
2. To meet the minimum requirements defined in Windows Hardware Compatibility (WHCP), an internal USB touch device is required to support Selective Suspend. For more information about how to enable Selective Suspend, please refer to the Selective suspend for HID over USB devices documentation.
3. In response to D2 or port suspend, the device firmware should enable Wake-on-Touch. Depending on the HID feature report for latency setting, the device firmware may choose different scan rates or gestures for Wake-on-Touch to achieve better user experience. For more information regarding the latency settings, please refer to the “USB Devices” section in the Touchscreen Power Management documentation.
4. Validate that the device wakes up on the appropriate touch gesture(s), based on the OEM requirements and/or form-factor of the device. Also validate that it does not cause spurious wake events and the power consumption from enabling the feature is acceptable. See Wake-on-Touch Validation Guide.

> **Note**
>
> If the USB touch is implemented as a USB function of a USB 3.0 multi-function device, it should support “function suspend” so the touch function can enter a low-power state for Wake-on-Touch independently of other functions. The Microsoft-provided parent driver (Usbccgp.sys) supports function suspend and remote wake-up features.

INF Requirements

The HIDClass Wake-on-Touch power policy is not opted-in by default.

For the touch controllers that use inbox drivers (HIDUSB/HIDI2C/HIDSPI), OEMs/IHVs should create an INF, and add the following to its DDInstall.HW section to opt-in the Wake-on-Touch policy.

```
Include = input.inf
```

```
Needs = WakeScreenOnTouch.HW
```

The Selective suspend for HID over USB devices MSDN article gives an example of how an OEM/IHV INF references the inbox INPUT.INF to enable a different feature named Selective Suspend. OEMs/IHVs may do similar in their INF but referencing the section mentioned above to enable Wake-on-Touch.

To simplify this INF, OEMs/IHVs may also consider creating an Extension INF instead. For more information regarding Extension INF, please refer to the Using an Extension INF file documentation.

If the device has opt-ed in any other HID policy that conflicts with the Wake-on-Touch policy, either of the policies will be ignored.

If the device has multiple HID collections including touch screen collections (UsagePage:0xD, Usage:0x4) and non-touch-screen collections, the touch firmware should only support wake from the touch screen collections. If a non-touch-screen collection may signal wake too, the device will be still brought back to D0 to drain the power.

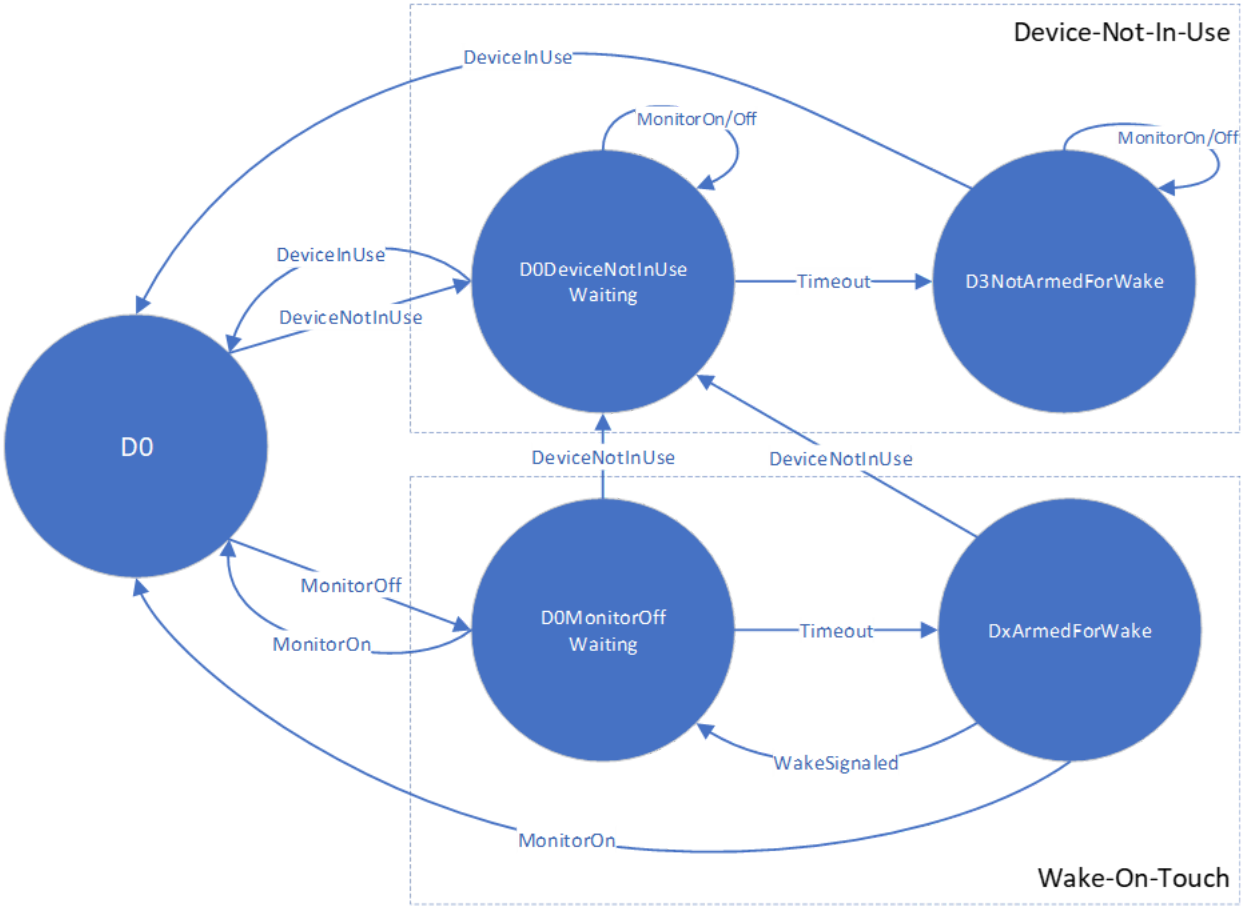
Enabling Wake-on-Touch With a Custom HID Mini Driver

Article • 10/05/2021

This topic discusses how to implement Wake-on-Touch with a custom HID mini driver.

Background: HIDClass Power Policy for Wake-On-Touch

The following state diagram illustrates the logical states the input device will be placed into when it supports the Wake-on-Touch power policy.



Notes:

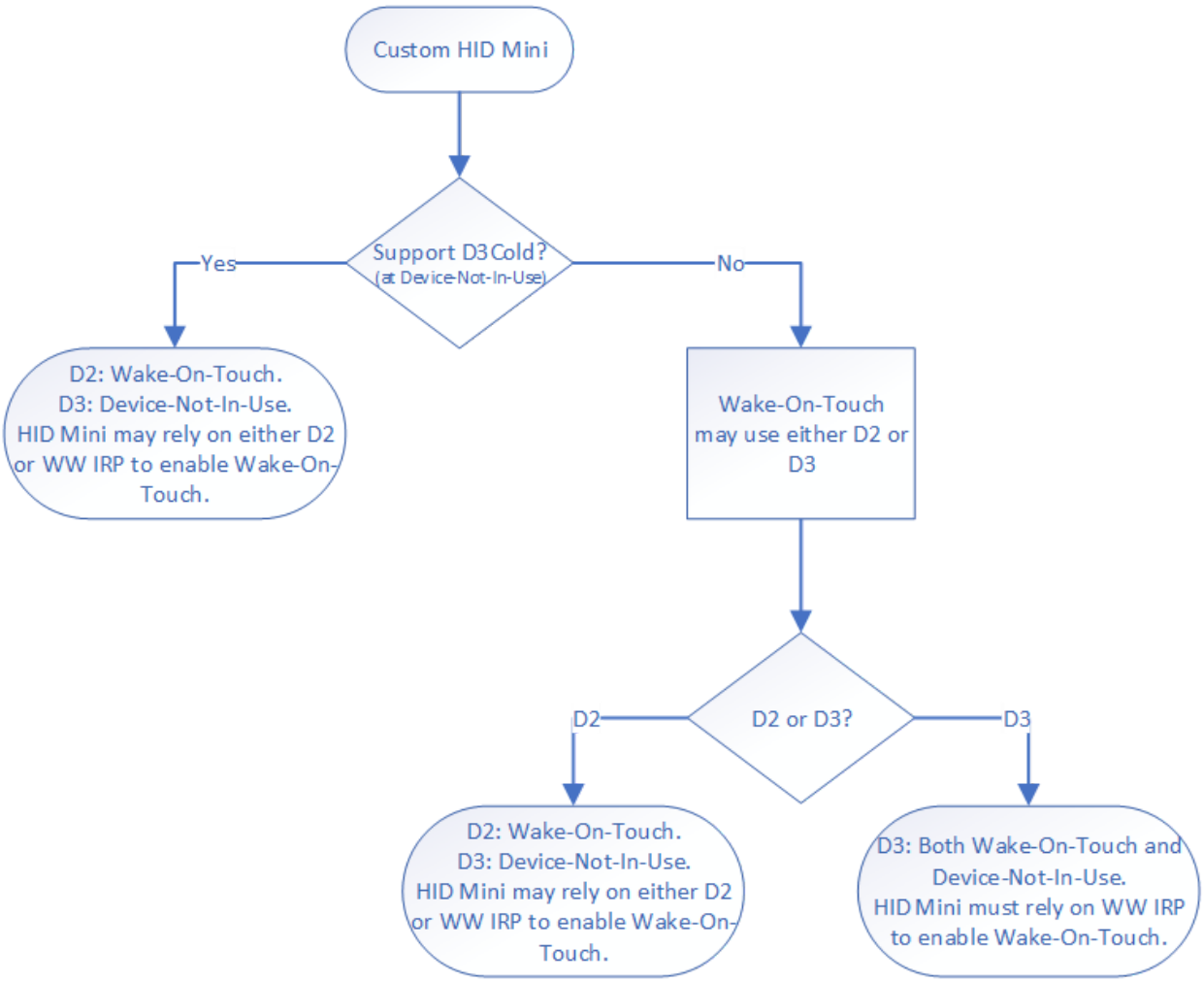
1. Dx = DEVICE_CAPABILITIES.DeviceWake
2. The idle timer interval is 1 second

How to Enable Wake-on-Touch

1. Decide the D-State for Wake-on-Touch and implement the custom HID mini driver with the guidance in “CHoose D-State for Wake-on-Touch” section below for some guidance.
2. Build an INF to opt-in the Wake-on-Touch policy as per the “INF Requirements” section.
3. Validate that the device wakes up on the appropriate touch gesture(s), based on the OEM requirements and/or form-factor of the device. Also validate that it does not cause spurious wake events and the power consumption from enabling the feature is acceptable. See Wake-on-Touch Validation Guide.

Choose D-State for Wake-on-Touch

For custom HID mini driver, the following diagram has the guidance about how to choose between D2 and D3. Assuming the touch controller needs the main device power source in its Wake-on-Touch mode, when D3Cold is supported, D2 is recommended for Wake-on-Touch, as D3Hot may be followed by D3Cold where Wake-on-Touch is not possible.


Custom HID Mini
Support D3Cold?
(at Device-Not-In-Use)
Yes
No
D2: Wake-On-Touch.
D3: Device-Not-In-Use.
HID Mini may rely on either D2 or WW IRP to enable Wake-On-Touch.
Wake-On-Touch may use either D2 or D3
D2 or D3?
D2
D3
D2: Wake-On-Touch.
D3: Device-Not-In-Use.
HID Mini may rely on either D2 or WW IRP to enable Wake-On-Touch.
D3: Both Wake-On-Touch and Device-Not-In-Use.
HID Mini must rely on WW IRP to enable Wake-On-Touch.

INF Requirements

The HIDClass Wake-on-Touch power policy is not opted-in by default.

For the touch controllers that use custom HID mini drivers and INFs, OEMs/IHVs should create an INF, and add the following to its DDInstall.HW section to opt-in the Wake-on-Touch policy.

```
Include = input.inf
```

```
Needs = WakeScreenOnTouch.HW
```

If the device has opted in any other HID policy that conflicts with the Wake-on-Touch policy, either of the policies will be ignored.

If the device has multiple HID collections including touch screen collections (UsagePage:0xD, Usage:0x4) and non-touch-screen collections, the touch firmware should only support wake from the touch screen collections. If a non-touch-screen collection may signal wake too, the device will be still brought back to D0 to drain the power.

Understanding When to Enable or Disable Wake-on-Touch

In Wake-on-Touch scenarios, HIDClass requests IRP_MN_WAIT_WAKE to arm the device for wake before requesting D-IRP to power down the touch device. In the Device-Not-In-Use scenario, HIDClass requests D-IRP directly to power down the touch device without requesting IRP_MN_WAIT_WAKE first.

- If the custom solution uses D2 for Wake-on-Touch and uses D3 for Device-Not-In-Use, it may simply rely on D-state in the D-IRP to differentiate between two scenarios, and only enable Wake-on-Touch on D2.
- If the custom solution uses D3 in both scenarios, it must rely on the presence of IRP_MN_WAIT_WAKE to differentiate between two scenarios and enable Wake-on-Touch when getting D3 request after IRP_MN_WAIT_WAKE.

It’s recommended that the custom HID mini driver not rely on any other system notifications such as GUID_MONITOR_POWER_ON, since that may potentially cause timing issues with the HIDClass-requested IRP_MN_WAIT_WAKE and D2/D3.

Device Reset Notification

This is not a requirement specific to Wake-on-Touch, but a general requirement for HID mini drivers. The custom HID mini driver is responsible for completing IOCTL_HID_DEVICERESET_NOTIFICATION whenever it realizes the device state loss and

the need of device reinitialization. This may be implemented in the device-initiated reset handler, or in driver's D0Entry routine for example after it detects the resume from D3Cold.

Touchscreen Validation Guide

Article • 10/05/2021

This section is a validation guide for a Windows Touchscreen device, for Windows 10 and later operating systems.

The guidance provided here ensures compliance with Hardware Lab Kit (HLK) testing for a Windows Touchscreen device.

In this section

Topic	Description
Touchscreen Validation Overview	This topic presents a brief overview of the assumed device test conditions for Touchscreen device validation.
Hardware Requirements and Vendor Information	This topic provides a list of the test equipment that is required for the Touchscreen-related tests in the Windows Hardware Lab Kit (HLK), and information about the vendors who stock the listed equipment.
General Testing Guidelines	This topic provides general guidelines for testing a Windows Touchscreen device, using the Windows Hardware Lab Kit (HLK) for Windows 10.
Touchscreen Tests	This section lists the topics that provide details about the Touchscreen tests in the Windows Hardware Lab Kit (HLK) for Windows 10.
Simultaneous Pen and Touch Validation	This section provides details on validating simultaneous pen and touch reporting for Windows 10
Wake-on-Touch Validation Guide	This topic provides information about how to validate wake-on-touch functionality.
Appendix	The appendix for the Windows Touchscreen Validation Guide, provides information about common HID-specific error messages.

Touchscreen Validation Overview

Article • 01/18/2019

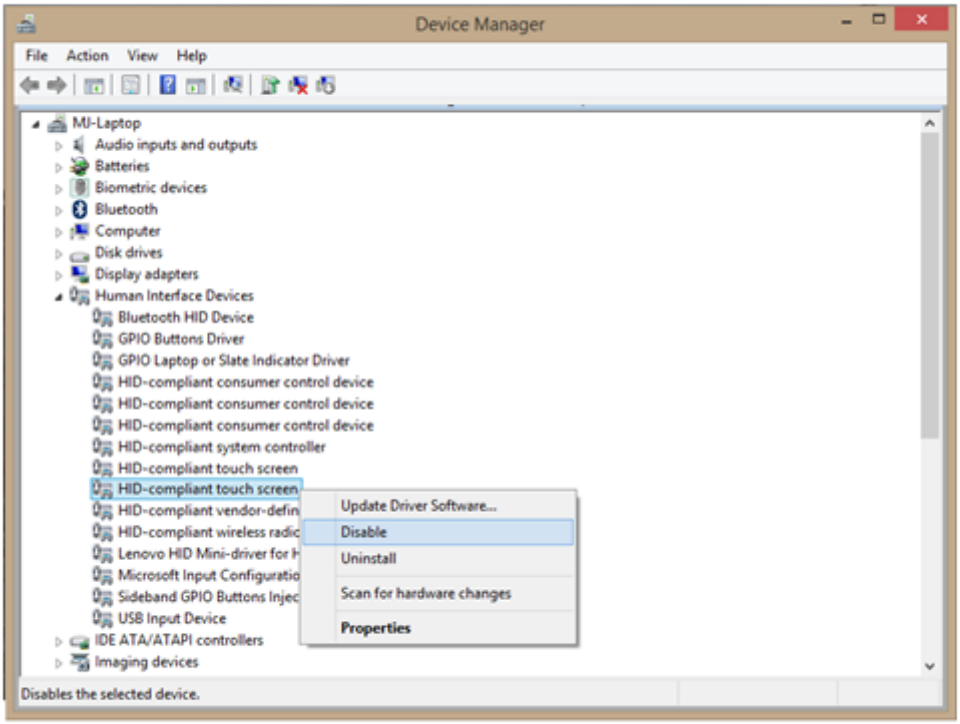
This topic presents a brief overview of the assumed device test conditions for Touchscreen device validation.

In order to ensure that all Windows Hardware Lab Kit (HLK) tests run successfully, it is important to follow the guidelines in this document as closely as possible.

This Touchscreen Validation Guide makes the following assumptions about the test environment of the device:

- HLK tests are being run on a device with Windows 10 installed.
  - Only the Windows 10 for desktop editions (Home, Pro, Enterprise, and Education) are supported by the HLK tests at this time.
- Device under test has a diagonal screen size of 4.5" - 30."
- Device reports HID usages as described in the Protocol Implementation section of the *Touchscreen Implementation Guide*.
- Testing setup has all the required equipment that is detailed in Hardware Requirements and Vendor Information.
- Each HLK test is performed according to the General Testing Guidelines, as well as any notes and instructions that are specific to that test.
- A test system that has multiple Touchscreen devices, has only one of them enabled - the one under test - and all others disabled. Use Device Manager to make this configuration, as shown in the following screenshot.

  Note The HLK tests will NOT work properly if a system has more than one touchscreen enabled!



If your device or testing setup does not meet the criteria outlined in the preceding list, you will not be able to successfully perform the Windows 10 HLK tests for your Touchscreen device.

Hardware Requirements and Vendor Information (touchscreen-hardware-requirements-and-vendor-information)

Article • 03/18/2021

This topic provides a list of the test equipment that is required for the Touchscreen-related tests in the Windows Hardware Lab Kit (HLK), and information about the vendors who stock the listed equipment.

Here's a list of all the required test equipment for completing the Touchscreen Tests in the HLK.

Precision Touch Test Tool (PT3)

You can use either one of these tools for precision touch testing.

- TRIEX/Tactile Precision Touch Test Tool (PT3)
- ITRI Precision Touch Test Tool (PT3)

Rotation and Acoustic (RA) Tool

You can use either one of these tools for RA testing.

- TRIEX/Tactile RA Tool
- ITRI RA Tool

Contacts for Tactile and ITRI

- Tactile Automation, Inc. Emery Parks +1 (425) 310-8380 emeryp@tactileautomation.com
- Industrial Technology Research Institute Sen Yih Chou +886 (03) 5743887 senyih@itri.org.tw

General Testing Guidelines (touchscreen-general-testing-guidelines)

Article • 03/18/2021

This topic provides general guidelines for testing a Windows Touchscreen device, using the Windows Hardware Lab Kit (HLK) for Windows 10.

There is a set of standards defined in the Windows Compatibility Requirements document for devices and systems that support Windows Touch. Both automated and manual tests are included in the HLK, to validate that the touch devices that support Windows Touch, meet the quality bar per the compatibility requirements. This section describes the process that you should follow before you start using the HLK to test a Windows Touchscreen device and complete the certification for the device.

- Some of the Windows Touchscreen tests, such as Touch UX (user experience), are expected to pass, mainly by using freehand operation.
- For point tests, align the contact device or devices with the required target points, and make contact on the digitizer in a movement that is perpendicular to the touch screen. To reduce issues of parallax, place the eye directly behind the contact device so that the eye, contact device, and target point are all in a line perpendicular to the screen. For the press and hold tasks, keep the contact device steady while you touch the screen.
- For the line test, follow the point test guidance when you begin and end the line trace at the indicated start and end points. Trace the line as closely as possible to the target line. Accurate parts of the traced line appear in green, and inaccurate portions appear in red.
- Unless otherwise specified, the device should always have AC power when you perform the HLK tests.
- Unless otherwise specified, all tests should be run with "Test Signing Mode" set to ON. To do this, perform the following steps:

    1. Open a **Command Prompt** window as an Administrator.
    2. Enter this command: **bcdedit -set testsigning on**
    3. Restart the computer.

- Unless otherwise specified, use 7mm diameter contacts for tests requiring use of the PT3 (Precision Touch Testing Tool). For "thumb testing" use the 16mm contacts.

  You may also use a human finger for any tests that use the 7mm slugs, and a human thumb for any tests that use the 16mm slug.

  For devices that have a raised bezel or other physical consideration that prevents the usage of the approved testing equipment (7mm and 16mm slugs), it will be necessary to pass the tests using a human finger.

- The Windows 10 HLK only supports testing a single touch digitizer at a time. If your device exposes multiple touch TLCs, you will need to do the following to ensure that the correct touch digitizer is being tested:

  1. Open **Device Manager**, then expand the **Human Interface Devices** node.

  2. Find and disable all HID-compliant touch digitizers, with the exception of the one that you want to validate.

Touchscreen Tests

Article • 01/18/2019

This section lists the topics that provide details about the Touchscreen tests in the Windows Hardware Lab Kit (HLK) for Windows 10.

In this section

Topic	Description
Digitizer Jitter	This is to test the jitter characteristics of a Windows Touchscreen device (or digitizer).
Five Point Minimum	This is to test the five-point contact capabilities of a Windows Touchscreen device.
Buffering	This is to test the data buffering capabilities of a Windows Touchscreen device.
Input Separation	This is to test the touch input separation capabilities of a Windows Touchscreen device.
Noise Suppression	This is to test the ability of a Windows Touchscreen device to suppress invalid input (noise) that could otherwise be mistaken as touch input.
Touch Accuracy	This is to test the touch input reporting accuracy for a Windows Touchscreen device.
Reporting Rate	This is to test the contact reporting rate of a Windows Touchscreen device.
Touch Resolution	This is to test the resolution of a Windows Touchscreen device.
Touch Response Latency (Down)	This is to test the down latency behavior of a Windows Touchscreen device.
Touch Response Latency (Panning)	This is to test the response panning latency characteristics of a Windows Touchscreen device.

Topic	Description
Touch Test	This is to test the type of touch input that falls into the category of Custom Gestures for a Windows Touchscreen device. This test also checks the 3rd party driver dependencies for the Touchscreen device.
Touch UX	This is to test the nature of the user experience for a Windows Touchscreen device.
Hid Validation	This is to test the HID protocol capabilities of a Windows Touchscreen device.

Simultaneous pen and touch validation

Article • 06/21/2022

In previous versions of Windows, touch contacts were supressed when pen input was detected within range of the digitizer. Starting in Winodws 1607, devices are able to report simultaneous pen and touch contacts to applications. This topic describes the tools and applications for validating your solution on Windows systems.

Note: On Windows 11 only, before running latency tests, set the Bluetooth & devices > Touch > "Three- and four-finger touch gestures" setting to "Off" as this can introduce a small delay into the touch responsiveness.

Simultaneous pen and touch reporting requirements

Currently, the Windows Hardware Compatibility Program does not contain requirements for simultaneous pen and touch performance. Microsoft instead provides the following recommendations for simultaneous pen and touch performance.

- A device should continue to meet all Windows 10 Pen compatibility requirements when five simultaneous touch contacts are present on the screen.
- A device should continue to meet all Windows 10 touch compatibility requirements when a pen is within reporting range of the screen.

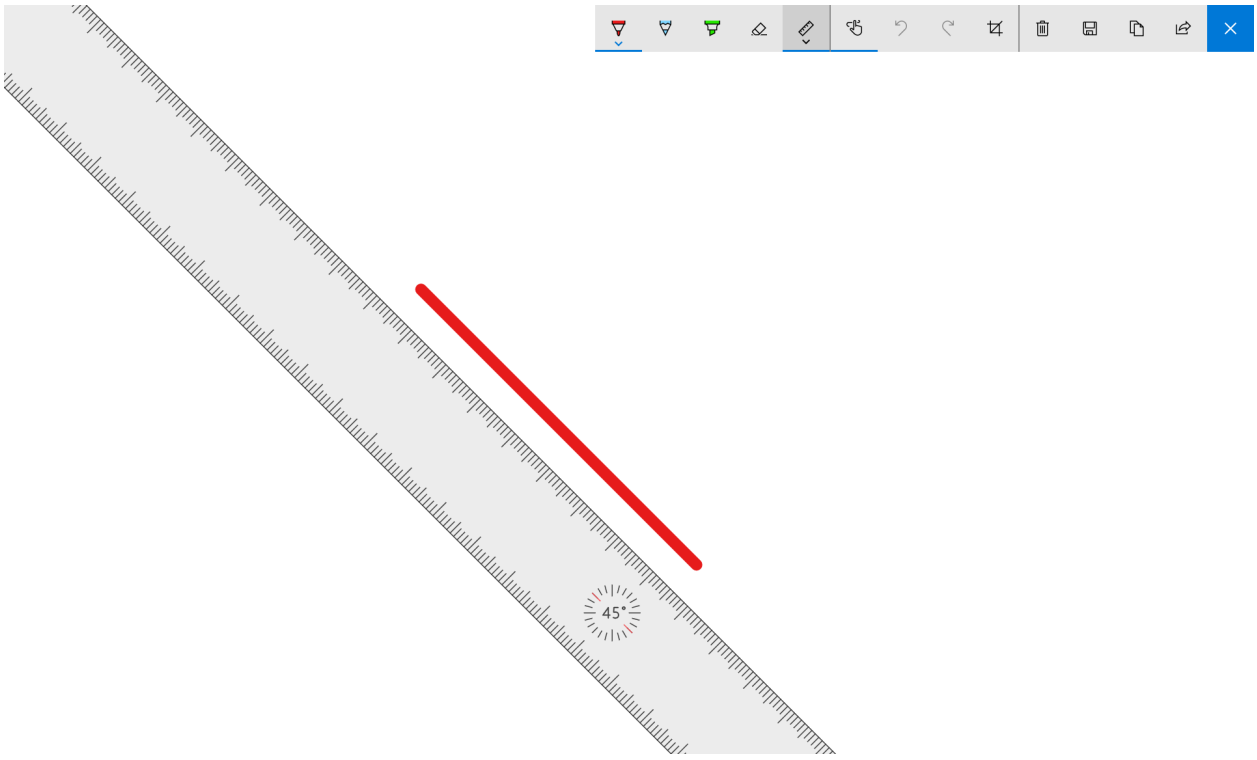
Devices capable of following these recommendations will be able to follow future requirements for pen and touch reporting

Validating Simultaneous Reporting Support

The first step in validating a Simultaneous Pen and Touch (SPT) system is to ensure the system supports sending touch and pen input simultaneously. Windows supports several ways of verifying this using basic applications. The two recommended methods are detailed below.

Windows Ink Workspace

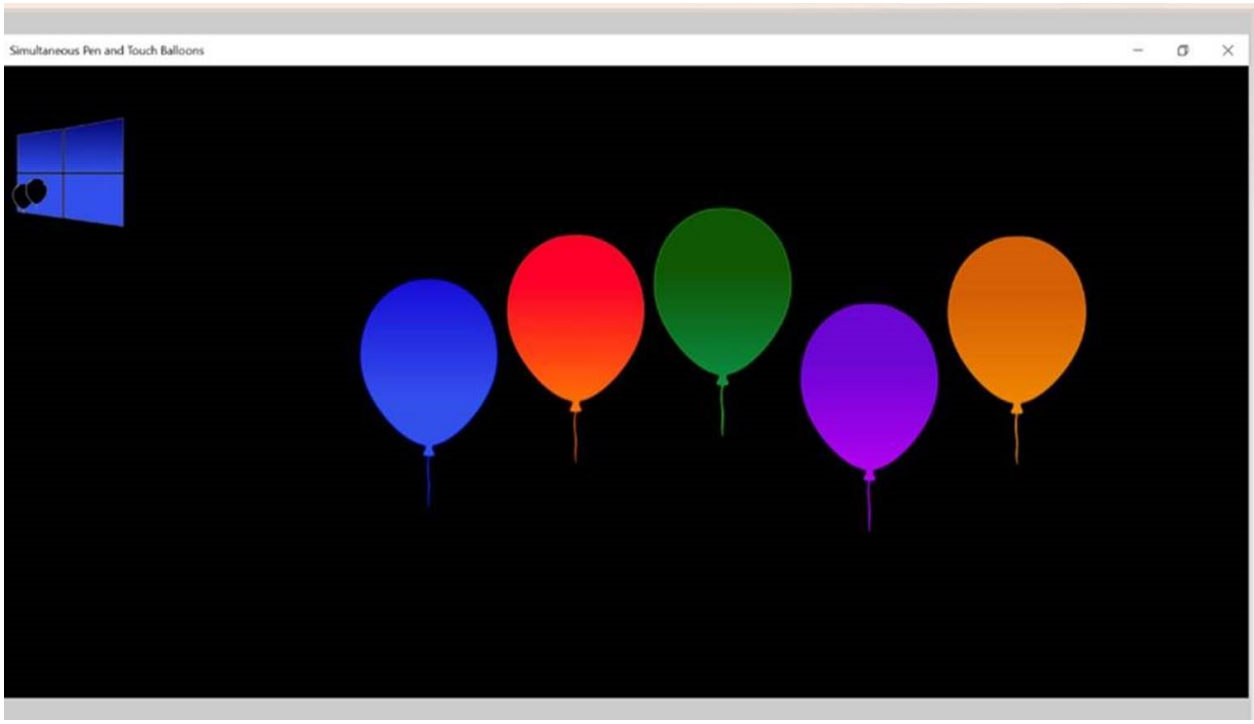
Using the Windows Ink Workspace and a pen device, you can validate an SPT system by using the workspace ruler.



1. Using touch, grab the Ruler and move its location on the screen.
2. Without lifting the touch contact from the screen, bring the pen into contact with the screen on the canvas, then remove the pen contact.
3. If the system supports SPT, you should still be able to move the Ruler with the touch contact that is still against the screen. If the system does not support SPT, you will need to remove the touch contact from the screen and tap the Ruler again to move the Ruler.

Simultaneous pen and touch baloons

You can also use the Simultaneous Pen and Touch Balloons App developed by Blue Lan to determine if a system supports SPT. The app renders several floating balloons on the screen.



To validate SPT on the system, place a finger on a balloon to stop it, and then tap the stopped balloon with your pen. If balloon is popped by the pen, SPT is supported on the system. If the balloon moves or is not popped, SPT is not supported by the touch device's firmware.

Validating pen and touch with the Windows Hardware Lab Kit

To ensure that a device can meet Microsoft's Pen and Touch requirements, Microsoft provides pen and touch tests as part of the Windows Hardware Lab Kit (HLK). Some of these tests can be used to verify that a device meets Microsoft's SPT requirements, as many of the tests now accept both forms of input. The following sections provide detailed guideance on these tests and test proceedures.

> **Note**
>
> Testing simultaneous pen and touch inputs with the HLK requires HLK build 14971 or greater

Pre-testing requirements

In addition to the setup steps required for Touch HLK testing, testing SPT requires you to download and have two additional .reg files:

- Run_Before_SPT_Testing_And_Restart.reg Pen and Touch Validation reg files/Run_Before_SPT_Testing_And_Restart.reg)
- Run_After_SPT_Testing_And_Restart.reg Pen and Touch Validation reg files/Run_After_SPT_Testing_And_Restart.reg)

Be sure to run the appropriate .reg file before SPT testing to enable the correct testing conditions, and after SPT testing to return the system to its normal state.

Validating pen requirements

Ensuring a device meets the SPT pen requirements while touch is present on the system can be performed using the Pen HLK. This validation can be performed by running the Pen HLK tests through HLK Studio or manually.

Windows supports SPT testing on the following Pen HLK tests when performed with five fingers in contact with the screen:

- Buffering
- Edge Gutter
- Resolution
- Moving Jitter
- Hover Accuracy
- Hover Range

For more information on these tests, see the Pen validation guide

Finger placement

To ensure that finger placement does not interfere with the results of the Pen HLK tests, do not place finger contacts in the green and yellow starting/ending sections on a test iteration, in the path of the pen.

For all tests, place all finger contacts against the screen before bringing the pen in range of the screen. Fingers can be moved during the test, but do not lift fingers from the screen until the end of an iteration. Additionally, be careful of bringing touch contacts near the buttons used for controlling test iterations.

Validating touch requirements

Ensuring a device meets the SPT pen requirements while a pen is present on the system can be performed using the Touch HLK. This validation can be performed by running the

Touch HLK tests through HLK Studio or manually.

Windows supports SPT testing on the following Touch HLK tests when performed with the pen in contact with the screen:

- 5 Point Minimum
- Buffering
- Latency - Down
- Reporting Rate

For more information on these tests, see the Touchscreen validation guide

Pen Placement

To ensure the pen placement does not interfere with the results of the Touch HLK tests, place the pen as far from the starting/ending sections, and the path of the fingers as possible. If a finger passes within 12cm of the pen during an iteration, Windows' built in palm rejection may choose to suppress the contact, causing the iteration to fail.

To decrease the likelihood of touch contacts being suppressed, you can verify the system's handedness setting. To do this, open the **Settings** menu, and select **Devices**. On the **Devices** list on the left, select **Pen & Windows Ink**.

If the **Choose which hand you write with** option is set to "Right Hand", placing the pen in the bottom right corner of the screen when a touch target is not in that area will greatly reduce the chance of contacts being suppressed. If it is set to "Left Hand", placing the pen in the bottom left corner of the screen will do the same.

For all tests, bring the pen in contact with the screen before bringing any touch contacts against the screen. The pen can be moved during the test, but do not lift the pen from the screen until the end of an iteration. Be careful when bringing the pen near the buttons used for controlling test iterations, as the pen may interact with them.

Additional testing tools

The test tool Digiinfo can be used to view input from touch and pen devices in real time to inspect issues with input reporting. For SPT validation, Digiinfo can be used simply to validate that input is being received from all five fingers when pen is in range of the screen, that the report rate of each touch contact equals/exceeds 60Hz, and that the report rate of the pen is at least 133Hz.

Digi Info

File Collection View Help

Digitizer Type: Integrated Touch, Integrated Pen, Multi Input, Ready

Sampling Rate: 137 100 100 100 100 100

Server Connection: None

Start Stop Clear Packets

Timestamp	ScanTime	X	Y	Cent...	Cent...	Width	Height	ContactId	ActualCount	Message	Buttons And Flags
360370515	24949	746	756					0	1	WM_INPUT	Confidence Tip
360370515	36300	3066	3146	3066	3146	458	505	132	5	WM_INPUT	Confidence Tip
360370515	36300	2769	5784	2769	5784	611	336	131	5	WM_INPUT	Confidence Tip
360370515	36300	545	3152	545	3152	359	505	130	5	WM_INPUT	

Digitizer Jitter

Article • 01/18/2019

This is to test the jitter characteristics of a Windows Touchscreen device (or digitizer).

Test name

- DigitizerJitterOneContact.json
- DigitizerJitterFiveContacts.json

Associated compatibility requirements

Device.Input.Digitizer.Touch.Jitter **Test purpose**

Verifies that a Windows Touchscreen device does not exceed the allowable amount of jitter – moving and stationary. **Tools required**

PT3 **Running the test**

Clean input is important throughout the system and across applications. Whether a user is performing a hold gesture to learn more about a user interface (UI) element, playing a game, or holding a key on the software keyboard, this fidelity allows software to correctly interpret a user's intention and provide an optimal touch experience.

This test presents twelve distinct scenarios that examine jitter during static contact and linear motion. Most of these interactions require the use of assistive technology.

Command syntax

Logo3.exe -config DigitizerJitter.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Five Point Minimum

Article • 01/18/2019

This is to test the five-point contact capabilities of a Windows Touchscreen device.

Test name

5TouchPointMinimum.json **Associated compatibility requirements**

- Device.Input.Digitizer.Touch.MinContactCount
- Device.Input.Digitizer.Base.HIDCompliant

Test purpose

Verifies that a Windows Touchscreen device supports a minimum of five simultaneous touch inputs. **Tools required**

None. Can optionally use PT3. **Running the test**

Because touch capabilities must be consistent across all computers, a minimum number of supported touch points are required. Five-finger support allows for whole-hand interactions in addition to quick and efficient touch-screen typing.

This test presents a single scenario that asks you to briefly place five or more contacts on the screen. All contacts that are placed at the beginning of the test must continue to report for the full duration of the test.

Command syntax

Logo3.exe -config 5TouchPointMinimum.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Buffering (touchscreen-buffering)

Article • 03/18/2021

This is to test the data buffering capabilities of a Windows Touchscreen device.

Test name

Buffering.json **Associated compatibility requirements**

Device.Input.Digitizer.Touch.Buffering **Test purpose**

Verifies that a Windows Touchscreen device supports the required amount of buffering that prevents data loss. **Tools required**

None. **Running the test**

Because some touch events can be lost while the digitizer is waking up, it is necessary to test that the touch buffer correctly stores data. The test also ensures that the user experience is not degraded by data loss, for example, losing an edge swipe.

Launch the test and follow the instructions. When you swipe the screen, the test will check to see if enough touch data was registered. If enough data was registered, then the test will pass. Please ensure that the swipe is quick – no longer than half a second.

Command syntax

Logo3.exe -config Buffering.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Input Separation

Article • 01/18/2019

This is to test the touch input separation capabilities of a Windows Touchscreen device.

Test name

InputSeparation.json **Associated compatibility requirements**

Device.Input.Digitizer.Touch.FingerSeparation **Test purpose**

Verifies that a Windows Touchscreen device meets the requirements for zoom, and multi-finger tap and swipe motions. **Tools required**

PT3 **Running the test**

Users often place their fingers close together in fast-paced games, drawing applications, musical instrument applications, and other common applications. In addition, Windows supports a system of touch gestures that let users who have low vision, fully explore the user interface (UI) and accomplish common tasks. Much of this gesture language relies on multiple fingers in close proximity.

This test consists of five scenarios that enforce the requirement that contacts in close proximity must be recognized separately. Broadly, this test verifies that a device meets these requirements during horizontal, vertical, and diagonal motion, and during a pinch gesture.

Command syntax

Logo3.exe -config InputSeparation.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Noise Suppression

Article • 01/18/2019

This is to test the ability of a Windows Touchscreen device to suppress invalid input (noise) that could otherwise be mistaken as touch input.

Test name

NoiseSuppression.json **Associated compatibility requirements**

Device.Input.Digitizer.Base.ContactReports **Test purpose**

Verifies that a Windows Touchscreen device does not report data for locations where touch input does not occur. **Tools required**

PT3 **Running the test**

This test can help to ensure that touch systems can effectively tolerate common environmental conditions.

This test presents two scenarios: the only distinction between the two scenarios is the power source. For the Battery portion of the test, you should remove the external power source from a mobile system. For AIOs and external monitors, perform the test with AC power. In each case, the test validates that a contact that is held at the minimum required distance from the screen does not register any touch contacts.

Command syntax

Logo3.exe -config NoiseSuppression.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Touch Accuracy

Article • 01/18/2019

This is to test the touch input reporting accuracy for a Windows Touchscreen device.

Test name

PhysicalInputPosition.json **Associated compatibility requirements**

- Device.Input.Digitizer.Touch.Accuracy
- Device.Digitizer.Touch.HIDCompliantFirmware

Test purpose

Verifies that a Windows Touchscreen device reports all inputs that meet the following criteria: o Inputs within +/- 1 mm of the center of the physical input for all touchable areas outside 3.5mm of the digitizer's edge. o Inputs within +/-2 mm within 3.5 mm of the digitizer's edge. **Tools required**

- PT3
- 16mm slug, or a human thumb for running the thumb tests.
- Free hand for running the edge tests.

Running the test

Minimal offset between the actual and the reported point of contact, is a primary factor in real and perceived system accuracy.

The Touch Accuracy Test suite includes tests for linear accuracy (offset) and edge performance, totaling seven scenarios. For the Edge (Inward) scenario, speed is measured and enforced. The tool warns you when motion is too slow, and then asks you to repeat the interaction. Note that moving too slowly does not generate a failure.

Command syntax

Logo3.exe -config PhysicalInputPosition.json **Passing criteria**

95% of the iterations must pass in order to complete with passing status.

Reporting Rate

Article • 01/18/2019

This is to test the contact reporting rate of a Windows Touchscreen device.

Test names

- ReportingRateOneContact.json
- ReportingRateFiveContacts.json
- ReportingRateMaxContacts.json

Associated compatibility requirements

Device.Input.Digitizer.Touch.ReportRate **Test purpose**

Verifies that a Windows Touchscreen device correctly reports physical contact at a rate greater than, or equal to the display refresh rate. **Tools required**

PT3 **Running the test**

A packet rate that is equal to, or greater than the display rate will ensure that no visual glitches appear when using the touchscreen. A high packet rate helps ensure smooth, fluid motion, and it supports the feeling that the user interface (UI) is connected to the user's finger. The input rate data and the increased data throughput both improve gesture recognition.

This test includes four separate reporting rate scenarios that are distinguished by single and multiple contacts and stationary contact versus motion. In each scenario, the test validates that the minimum reporting rate is maintained for each contact (and specifically not just the average reporting rate).

Command syntax

Logo3.exe -config ReportingRate.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Touch Resolution

Article • 01/18/2019

This is to test the resolution of a Windows Touchscreen device.

Test name

TouchResolution.json **Associated compatibility requirements**

Device.Input.Digitizer.Touch.Resolution **Test purpose**

Verifies that a Windows Touchscreen device has a resolution that is greater than, or equal to the native display resolution. **Tools required**

None. **Running the test**

To support the assertion that every pixel is touchable, the resolution of the touch screen must be at least as high as the resolution of the display. This test verifies that capability. To complete this test, simply touch the screen. The program validates that the digitizer that reports the contact has a resolution that is at least as high as the connected display.

Command syntax

Logo3.exe –config TouchResolution.json **Passing criteria**

100% of the iterations must pass in order to complete with passing status.

Touch Response Latency (Down)

Article • 09/03/2021

This is to test the down latency behavior of a Windows Touchscreen device.

Test name

audiotouch.exe **Associated compatibility requirements**

Device.Input.Digitizer.Touch.Latency **Test purpose**

Verifies that a Windows Touchscreen device meets the response down latency requirements when the digitizer is active. **Tools required**

RA tool. See Hardware Requirements and Vendor Information. **Running the test**

Low response *latency* plays a large role in ensuring fast and fluid motion. This is similar to the Reporting Rate test, where a high packet rate helps ensure smooth, fluid motion. Conversely, a high degree of response latency breaks the illusion that the user is manipulating a physical object.

Note: On Windows 11 only, before running this test, set the Bluetooth & devices > Touch > "Three- and four-finger touch gestures" setting to "Off" as this can introduce a small delay into the touch responsiveness.

This test uses specialized hardware to measure the latency in touch packet delivery for touch-down. For more detailed instructions, see How to measure Touch Down Latency by using an Acoustic Measurement Tool.

Only 50 taps are required during the collection phase.

Command syntax

audiotouch.exe **Passing criteria**

Average latency recorded during the test must meet the requirement to complete with passing status.

Touch Response Latency (Panning)

Article • 09/03/2021

This is to test the response panning latency characteristics of a Windows Touchscreen device.

Test name

stepmotor.exe **Associated compatibility requirements**

Device.Input.Digitizer.Touch.Latency **Test purpose**

Verifies that a Windows Touchscreen device meets the response panning latency requirement. **Tools required**

RA tool. See Hardware Requirements and Vendor Information. **Running the test**

Low response *latency* plays a large role in ensuring fast and fluid motion. This is similar to the Reporting Rate test, where a high packet rate helps ensure smooth, fluid motion. Conversely, a high degree of response latency breaks the illusion that the user is manipulating a physical object.

This test uses specialized hardware to measure the latency in touch packet delivery for panning. For more detailed information, see How to measure Touch Panning Latency.

Note: For Windows 10 testing, please use the 7mm contacts instead of the 9mm contacts.

Note: On Windows 11 only, before running this test, set the Bluetooth & devices > Touch > "Three- and four-finger touch gestures" setting to "Off" as this can introduce a small delay into the touch responsiveness.

Also note that the Touch panning latency test is designed to evaluate device latency during active use on the touchscreen. The test utilizes the acoustic latency measurement tool that requires calibration to complete the test.

On USB systems that have selective suspend enabled, this calibration period may cause errors in the device due to wake times. For the purpose of this test, selective suspend may be disabled to ensure the device is in active mode for the full duration of the calibration and validation.

It is important to note that selective suspend must be re-enabled after the operator is done running the panning latency test since the rest of the test in the suite will not correctly evaluate the device with selective suspend disabled.

Command syntax

stepmotor.exe **Passing criteria**

Average latency recorded during the test must meet the requirement to complete with passing status. There is a 20% tolerance on panning latency tests.

Touch Test

Article • 09/17/2021

This is to test the type of touch input that falls into the category of Custom Gestures for a Windows Touchscreen device. This test also checks the 3rd party driver dependencies for the Touchscreen device.

Test names

Other.json **Associated compatibility requirements**

- Device.Input.Digitizer.Touch.CustomGestures
- Device.Input.Digitizer.Base.ThirdPartyDrivers
- Device.Input.Digitizer.Base.HIDCompliant

Test purpose

Verifies that a Windows Touchscreen device meets the custom gestures and third party driver requirements. **Tools required**

None.

Running the test

The Windows Touch Test combines a set of simple manual tests. The requirements that are tested, include the absence of third party drivers, the absence of custom touch gestures, and the allowed physical dimensions of the device.

Note: If wake-on-touch is enabled, answer "No" to the "Is the following input device on an in-box supported bus?" question, even if the input device is on an in-box supported bus. Answering "No" under any other cases when on an in-box supported bus is a violation of policy.

Command syntax

Other.json **Passing criteria**

- 100% of the iterations must pass in order to complete with passing status.

Touch UX

Article • 01/18/2019

This is to test the nature of the user experience for a Windows Touchscreen device.

Test name

UXTests.json **Associated compatibility requirements**

- Device.Input.Digitizer.Base.ContactReports
- Device.Input.Digitizer.Touch.Accuracy
- Device.Input.Digitizer.Touch.FingerSeparation
- Device.Input.Digitizer.Touch.Jitter
- Device.Input.Digitizer.Touch.ReportRate

Test purpose

Validates the Windows Touchscreen user experience that is associated with the hardware requirements. The tests are designed to run freehand.

The test ensures that all UI elements in the system can be accurately targeted by users with touch at the given resolution and scaling factor of the screen. The smaller size of the hitboxes in this test is by design as it accurately represents the resolution of the application. If the screen is super high resolution to the point where a 40px touch target (the size of the tile) is very small, then the tester needs to be able to hit the targets.

Note The test is using DIPS, not physical pixels, so make sure that the test is running on a system with an appropriate scale factor for the display. If the test is run at 100% scaling, the test will be more difficult.

Tools required

None. Manual test. **Running the test**

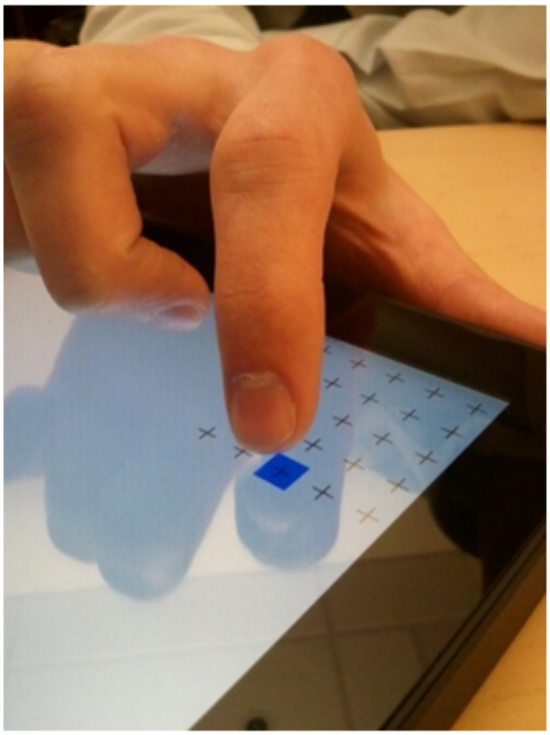
The Windows Touch UX Test suite combines disparate touch requirements into everyday tasks. The principle benefit of this test suite is to test the requirements together and to verify that a user can reliably complete the core Windows touch interactions.

Here's a list of the tests in the suite, along with descriptions, points to note and some recommendations.

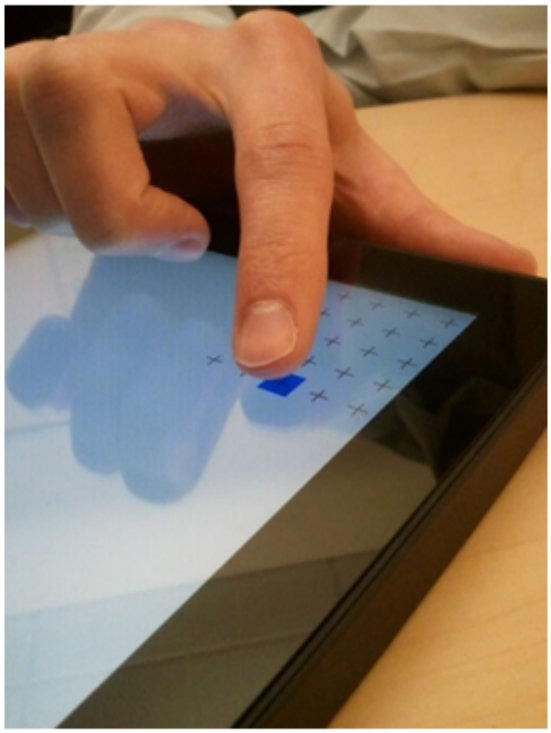
Tile Tap

Tile Tap is a simple accuracy test in the context of tapping tiles on the Search screen. It appears four times: one time in each screen quadrant. To complete the test, tap each blue box as it appears. We recommend that you use a *natural tapping* motion during your internal testing.

Here's an example of a precise tapping finger position.



Here's an example of the natural (recommended) tapping finger position.



Semantic Zoom

Semantic zoom is an active feature in many of the default controls in the Windows SDK. For example, you can see Semantic Zoom on the Start screen, where the user can pinch and zoom to reveal different views of the application tiles.

In each iteration of the test, you are asked to zoom in or zoom out. You can accomplish these, respectively, by pulling your fingers apart or by pulling them together. There are two versions of this test: one that uses two fingers and one that uses five fingers.

Quick Toss

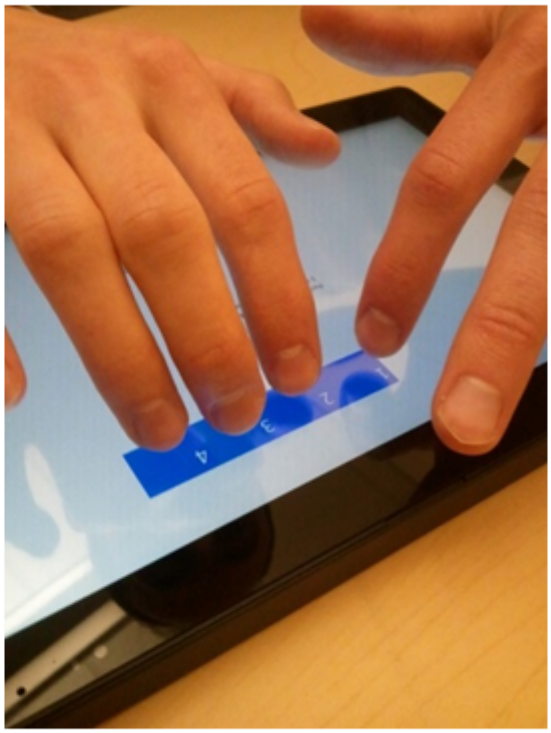
Quick toss is a punctuated panning motion that scrolls quickly through pages of content. The Quick Toss test goes through a series of iterations. In each iteration, you quickly swipe up or down the screen by using one or two fingers. (Note that speed is enforced in this test.) A quick toss that is performed too slowly will not cause a failure, but it will require you to repeat that iteration.

Note When inertia is applied and the tester cannot pan the color screen accordingly (that is, a faster swipe does not make color screens scroll faster), this test should be treated as a valid fail because the user cannot pan smoothly by using the desired velocity.

Keyboard

The Keyboard test simulates quick typing on the touch keyboard. It presents four keys that must all be tapped once within one second. (This represents a typing speed of four keys per second, or roughly 48 words per minute.)

The test times out one second after the first target is pressed. If a target is missed because of a dropped touch or a sequence of taps that are interpreted as a move, the test can appear to be lagging. The following image shows you how to place your fingers on the touch screen for this test.



Hyperlink

The Hyperlink test asks you to tap a series of hyperlinks that are rendered at their typical size. The test is complete after all of the hyperlinks have been tapped or missed.

Press and hold

Press and hold is a gesture that users can employ to discover more information about a UI element. For example, if you hold a finger down on the desktop, the context menu opens. The Press and Hold test presents four squares: three gray and one blue. You must press the blue square until it turns green. Repeat until the test is complete.

Mobar

Users can drag the dividing line between two active displayed applications, to change the area that is allocated to either application, or to remove one application from the

screen. The Mobar test validates that users can complete this activity by drawing a blue line, and then asking you to drag from any location on this line to a green box. The locations of the blue line and the green box change during the test, but the goal is always the same.

Backstack

The Backstack test models multitasking, whereby the user can pan left or right to reveal recently visited pages. In each iteration of the test, you must swipe left or right. You must complete each iteration to fulfill the test requirements.

Edgy

The Edgy test models the edge gestures that perform such actions as to display the **Action Center** or to cycle through and dismiss applications. To complete the test, swipe from the border area (or bezel) onto the visible screen area through the side of the screen that is colored blue.

Note Each swipe requires a minimum speed of 20mm/s. This is the same speed threshold that is used in the operating system. If the tester does not achieve this speed, an "Unknown Reason" error message will be displayed.

Select

The Select test presents a number of tiles, one of which is colored blue. The goal is to pull up or down on the blue tile toward the gray line without passing it. The test monitors for motion that is perpendicular to the direction of the select gesture; the user sees this motion as panning.

Rearrange

Rearrange is a gesture that is used to move application tiles on the screen. You accomplish this by applying a dragging motion to an application tile until it breaks free.

Command syntax

Logo3.exe -config UXTests.json

Hid Validation

Article • 01/18/2019

This is to test the HID protocol capabilities of a Windows Touchscreen device.

Test name

HIDValidator.exe **Associated compatibility requirements**

Device.Input.Digitizer.Base.HIDCompliant **Test purpose**

Verifies that a Windows Touchscreen device meets the HID protocol requirement. **Tools required**

None. **Running the test**

This runs unattended.

Command syntax

HIDValidator.exe

Wake-on-Touch Validation Guide

Article • 05/26/2022

Please note: The only Wake-on-Touch gesture supported on Windows 11 is **single finger touch** and the valid tap duration is between 50ms to 500ms. This document references single-tap interactions, but you may choose to implement a double-tap instead. A double-tap is designed as two taps where the second begins no later than 300ms from completion of the first.

Test Prerequesites

Ensure your SUT (System Under Test) supports Modern Standby

Use **powercfg /a** to confirm that SUT supports Modern Standby (S0 Low Power Idle)

```
C:\WINDOWS\system32>powercfg /a
The following sleep states are available on this system:
    Standby (S0 Low Power Idle) Network Connected
    Hibernate
    Fast Startup
```

Enable trace logging

Trace logs will be used for Test Case 2 and 5. To enable capturing a trace log:

1. Open command prompt as administrator
2. Enter this command line to start a trace: **wpr -start power**
3. While recording, run through the steps/scenarios to put the system into Modern Standby for a given time
4. Enter this command line to stop and save the trace into an event trace log (ETL): **wpr -stop filename.etl**

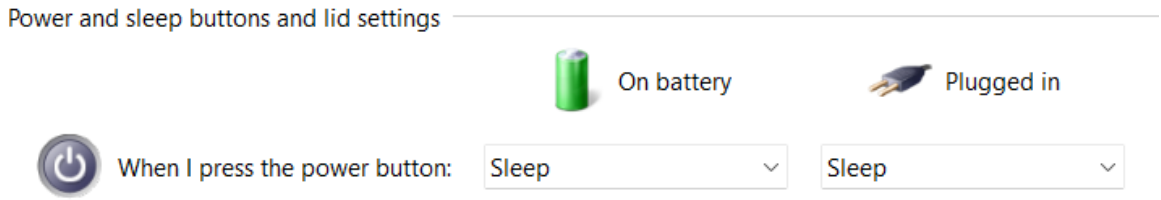
For more details of how to capture a log for issues that cannot be resolved by using SleepStudy you may refer to this link.

Test Cases

Follow the steps below to run tast cases for each scenario.

Test Case 1: Ensuring that SUT can be woken just after monitor off

1. Go to **Control Panel** > **Hardware and Sound** > **Power Options** > **Choose what the power buttons do** and ensure both **On battery** and **Plugged in** are set to **Sleep**



2. Put the device in AC power and press power button to put the device into sleep
3. Tap the screen with a single finger to wake the device
4. Make sure that device can be woken from sleep within 1 second
5. Repeat steps 2-4 for 5 times
6. Rerun the above steps but this time put the device in DC power and repeat steps 2-5

Expected results for Test Case 1

Successfully wake the SUT from sleep by tapping the screen with 1 finger (with 100% pass rate).

Test Case 2: Validate the SUT can successfully enter Modern Standby and validate wake reason is touch input

1. Ensure no other peripherals are attached to your SUT (Ex: USB thumb drive, mouse or ethernet dongles etc.)
2. In order to avoid network related interference, put the device in airplane mode
3. Put the device on DC power
4. Reboot your SUT so that the system is in a clean state (This step is to ensure no additional apps or updates running in the background)
5. Press power button for SUT to enter Modern Standby (display off)
6. Leave SUT in Modern Standby for at least > 10 minutes
7. Single tap to wake the monitor from Modern Standy (valid tap duration should be between 50ms to 500ms)
8. View wake reasons by opening the command prompt as an administrator and executing **powercfg /spr** to generate a SleepStudy report.
9. Open the html report and verify the enter and exit reason is **Input Touch**

> ⓘ **Note**

For more information on how to interpret sleep study reports, please see the **SleepStudy report page** or refer to expected result section below.

Expected results for Test Case 2

Look at the SleepStudy report and ensure that the device can successfully enter Modern Standby as well as getting good SWDRIPS and HWDRIPS with minimum wakes:

1. Verify that the system can achieve **>=80% SW DRIPS**
2. Verify there is **<=10% DRIPS delta** between **SW DRIPS** and **HW DRIPS (SW DRIPS – HW DRIPS)**

	START TIME	DURATION	STATE	ENERGY CHANGE		CHANGE RATE		% LOW POWER STATE TIME		% CAPACITY REMAINING AT START
14	10:55:45	0:14:54	Sleep	10 mWh	0% of battery	40 mW	Charge	SW: 88%	HW: 86%	77%

3. Verify there are **no unexpected wakes from touch input other than the intended one**, i.e. the sleep session ends at the time of intentional wake
4. Verify there are no intermediary sessions where the wake was due to a spurious / "ghost" touch

Check that the Modern Standby Exit Reason is due to touch input and the exit latency is **<=1 second**

5. Verify the exit reason specifies that it was due to a touch input in the sleep study report, as below:

	START TIME	DURATION	STATE	ENERGY CHANGE		CHANGE RATE		% LOW POWER STATE TIME		% CAPACITY REMAINING AT START
17	2021-09-28 10:55:45	0:14:54	Sleep	10 mWh	0% of battery	40 mW	Charge	SW: 88%	HW: 86%	77%

- Detailed Session Information

+ Battery Information

- General

Enter Reason	Transition To Sleep
Exit Reason	Input Touch
System GPU Adapter Count	1
Directed DRIPS Transition Count	0
Input Suppression Action Count	0
Power Button Suppression Action Count	0

6. Verify that the exit latency is <= 1 second

	START TIME	DURATION	STATE	ENERGY CHANGE		CHANGE RATE		% LOW POWER STATE TIME		% CAPACITY REMAINING AT START
17	2021-09-28 10:55:45	0:14:54	Sleep	10 mWh	0% of battery	40 mW	Charge	SW: 88%	HW: 86%	77%

- Detailed Session Information

+ Battery Information

+ General

- MS Exit Latency [ms]

Total Exit Time	326
Time Spent In Resiliency	90
Time Spent In Resiliency Notification	0
Time Spent In LPE	0
Time Spent In DAM	0
Time Spent In Maintenance	0
Time Spent In PLM	12
Time Spent In Shell	0
Time Spent In Connection	0
Time Spent Turning Display On	224
Graphics Device Interface (GDI)	0
Desktop Window Manager (DWM)	0
Monitor Power On	224
Request Processing	0

> **Note**
>
> If something is blocking the system to achieve SWDRIPS, you can visualize that in the SleepStudy html report by checking what is to blame and then take corrective action before running the next iteration. You can refer to **Top Offenders** section in the SleepStudy report as shown below.

Top Offenders

Top 5 Offenders, ranked by active time

NAME	TYPE	% ACTIVE TIME	ACTIVE TIME
BI	Activator	39%	0:05:54
NCSI	Activator	2%	0:00:23
Lock Screen	Activator	1%	0:00:13
Update Orchestrator Client	Activator	0%	0:00:06
Low Power Phase	PDC Phase	0%	0:00:05

Test Case 3: Validate touch is still functional as expected after waking from Modern Standby

Run the Windows Touch UX Test portion of the Touch HLK to test the following scenarios:

- Tile Tap (Top Left)
- Tile Tap (Top Right)
- Tile Tap (Bottom Left)

- Tile Tap (Bottom Right)
- Hyperlink Test
- Mobar Test
- Semantic Zoom (2 Fingers)
- Semantic Zoom (5 Fingers)
- Backstack
- Quick Toss Test
- Tile test (Select)
- Tile test (Rearrange)
- Press & Hold
- Edgy
- Keyboard

Expected results for Test Case 3

All test cases pass, as shown below:

Test cases		Result
Windows Touch UX Test	Tile Tap (Top Left)	Pass
	Tile Tap (Top Right)	Pass
	Tile Tap (Bottom Left)	Pass
	Tile Tap (Bottom Right)	Pass
	Hyperlink Test	Pass
	Mobar Test	Pass
	Semantic Zoom (2 Fingers)	Pass
	Semantic Zoom (5 Fingers)	Pass
	Backstack	Pass
	Quick Toss Test	Pass
	Tile test (Select)	Pass
	Tile test (Rearrange)	Pass
	Press & Hold	Pass
	Edgy	Pass
	Keyboard	Pass

Test Case 4: Validate a palm on the screen does not wake the device

1. Ensure no other peripherals are attached to your SUT (Ex: USB thumb drive, mouse or ethernet dongles etc.)
2. In order to avoid network related interference, put the device in airplane mode
3. Put the device on DC power
4. Reboot your SUT so that the system is in a clean state (This step is to ensure no additional apps or updates running in the background)
5. Press power button for SUT to enter Modern Standby (display off)
6. Leave SUT in Modern Standby for at least > 10 minutes
7. Place a palm on the SUT's screen

Expected results for Test Case 4

The SUT does not wake from Modern Standby when the palm is placed on the screen (i.e. the display should not light up and resume the system back to the active state).

Test Case 5: Validate there are no inadvertent wakes when the SUT's lid is closed

1. Shut the SUT's lid
2. Keep the lid closed for 30 minutes with the trace logging (refer to **Test Prequesites**) running in the background and ensure no wakes occurred in that time
3. View wake reasons by opening the command prompt as an administrator and executing **powercfg /spr** to generate a SleepStudy report
4. Open the html report to view the wake reason and exit latency
5. Open the trace logs to validate the device state is in Modern Standby and during this time no wakes occured

> ⓘ **Note**
>
> Check to see that lid state changes are handled properly (this can be verified in power traces / SleepStudy report) and the touch device is in the appropriate power state as the lid is opened and closed.

Expected results for Test Case 5

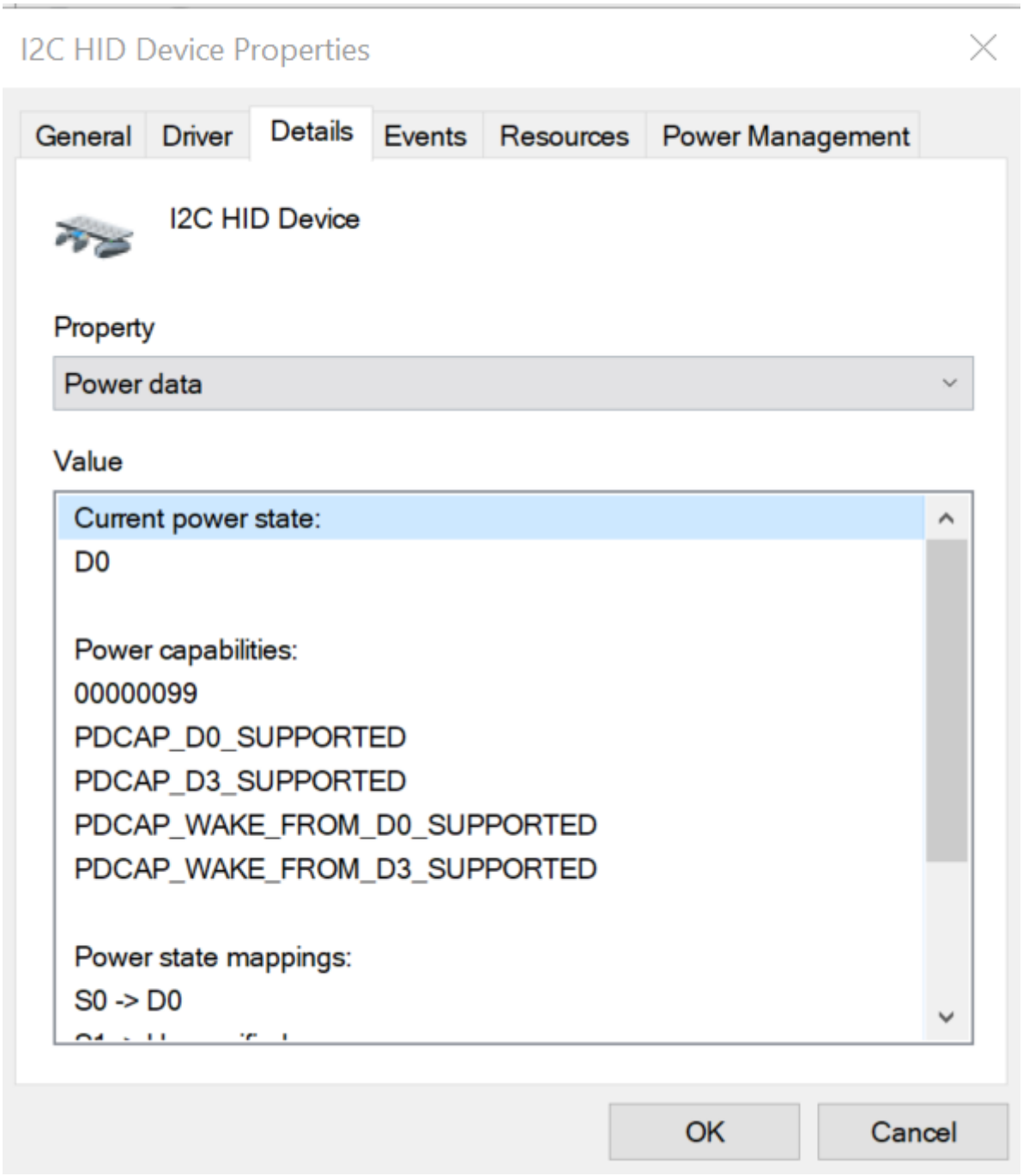
The wake reason in the SleepStudy report is **Lid** and no wakes occurred while the lid was closed.

Diagnostics

When Wake-on-Touch is not working as expected, here are some of the diagnostics approaches OEMs/IHVs may try before contacting Microsoft.

Power Capabilities in Device Manager

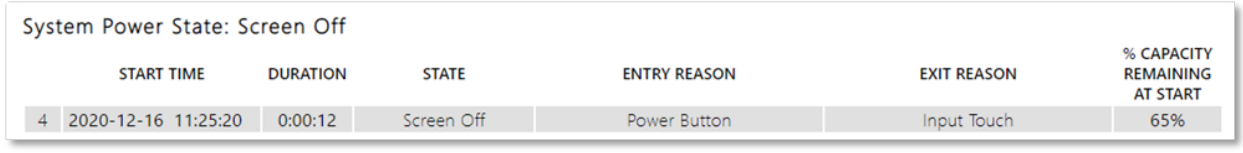
Some basic power capabilities can be examined in the “Power data” property in Device Manager.



Sleep Study (Wake Source)

By running this command, a sleep study report will be generated for the previous system state transitions. In the sleep study report, the “EXIT REASON” column tells if the system state is transitioned due to the touch input.

```
Powercfg.exe /spr
```



For more information about sleep study, please refer to the Modern standby SleepStudy documentation.

Trace Logs

In case further investigation is needed, trace logs can be captured for OEMs/IHVs to view or shared with Microsoft teams.

To capture trace logs, please use BusesTrace.cmd here and select "2) Input/HID components only" at its second menu.

Digitizer Display Mapping

Article • 05/20/2024

This section describes how a Windows touch or pen ditizer can be automatically mapped to the appropriate display, for Windows 10 and later operating systems.

The approach for automatically mapping the digitizer and its display differs for internal vs. external displays.

External displays

Windows will match external digitizers to their display using their Container ID configuration. Windows automatically generates a unique Container ID for the display by using the manufacturer ID, product ID, and serial number obtained from the Extended Display Identification Data (EDID). The display may override this Container ID by handling the DXGKDDI_GET_CHILD_CONTAINER_ID callback. The digitizer must then use this same Container ID, typically via its USB ContainerID descriptor.

If there is a single external display, the only requirement is that the digitizer is also marked as external - no other properties need to match. Matching by Container ID is only necessary when there are multiple external displays, even if not all of them are touch/pen-enabled.

> **Note**
>
> If the user has multiple touch/pen-enabled displays of the same make and model connected, it is important that each display has a different Container ID, to ensure that each of the digitizers is properly mapped to the appropriate display. If multiple devices share the same Container ID, Windows will be unable to map them. If the Container ID is derived from the EDID, then the EDID must have a unique serial number for every device.

Internal displays

If there is a single internal display, the only requirement is that the digitizer is also marked as internal - no other properties need to match.

When there are multiple internal displays, Windows will match digitizers to their display using their ACPI Physical Location Description (_PLD) configuration. In order for the

mapping to occur, the digitizer and display must match:

1. Their container IDs (typically this is the "internal" container ID)
2. Their _PLD **CabinetNumber** value (or if only _PLD v1 is supported, their **Lid** and **Dock** values)
3. Their _PLD **Panel** value

User overrides

Windows allows a user to override their digitizer's display mapping by opening Control Panel > Hardware and Sound > Tablet PC Settings, then clicking **Setup...**.

Related topics

Container IDs

Container ID support for displays

Using Microsoft OS Container ID Descriptors

DXGKDDI_GET_CHILD_CONTAINER_ID callback function

ACPI _PLD Specification

Appendix (touchscreen-appendix)

Article • 03/18/2021

The appendix for the Windows Touchscreen Validation Guide, provides information about common HID-specific error messages.

Here's a table of the HID-specific error messages related to the Windows Touchscreen device tests.

HID error #	Description
1	Invalid X bit size.
2	Invalid Y bit size.
3	Invalid packet transition.
5	Invalid X, Y.
6	Invalid width or height.
8	Invalid contact ID.
14	No data in frame.
16	Invalid # of contacts in frame.
17	More than max contacts in frame.
18	Sampling rate out of range.
21	Invalid actual count.
22	Invalid confidence.

This section provides information to help explain the error messages a bit further.

HID error #1

The bit count for Tx/Cx is outside the range [1,32]. Only checked if C is present.

HID error #2

The bit count for Ty/Cy is outside the range [1,32]. Only checked if C is present.

HID error #3

This can present itself with any of the following error messages:

- Last move location different. The coordinates of the "*tip switch clear*" report for a given contact are not the same as the coordinates of the "*last tip switch set*" report.
- Missing tip-on. The first report didn't have the tip switch set, or there were two packets in a row without the tip switch set.
- Missing tip. A contact present in the previously reported frame, with the tip switch set, was not found in the current frame.

HID error #5

This can present itself with any of the following error messages:

- (Invalid T) With C, Width, or Height present, Tx and/or Ty were not present or not within their logical range.
- (Invalid C) With T, Width, or Height present, Cx and/or Cy were not present or not within their logical range.
- (Invalid T/C combo) T was not contained within bounding box formed by C, Width, and Height.

HID error #6

This can present itself with any of the following error messages:

- "" Width and/or height found to be present, and either one was not present, or one/both were outside their logical range.
- (0) Width and/or height found to be present, but the logical value for one/both had a value of zero (0).

HID error #8

This can present itself with any of the following error messages:

- (Not present) The device does not support the *ContactID* usage in its descriptor.
- (Dupe in frame) A contact ID was duplicated in a single frame (sometimes caused by an incomplete frame being reported).

HID error #14

There were no contacts in the frame and the physical button is not down. This would imply that the physical button was up, but the physical button was not previously reported as being up.

HID error #16

The number of contacts in the frame did not match the reported *Actual Count*.

HID error #17

The number of contacts in the frame exceeded the maximum number of contacts that the device supports per *MAX COUNT*.

HID error #18

The sampling rate was not in the allowed range for the number of contacts reported.

HID error #21

The device does not support the *Actual Count* usage in its descriptor.

HID error #22

The confidence switch was not set (and the test was not the confidence test).

Legacy Touchscreen, Touchpad, and Pen Resources

Article • 01/24/2024

The topics in this section present legacy resources for Windows Touchscreen and Pen devices. This information applies to Windows 8 and earlier operating systems.

In this section

Expand table

Topic	Description
Windows Pointer Device Data Delivery Protocol	This section provides information about Windows 8 devices that support the pen, or touch, functionality.
Legacy Windows Touch Drivers	This section presents information about how to implement support for Windows Touch devices in Windows 7 and earlier operating systems.
Additional Resources	This topic presents resources for the legacy Windows Touchscreen and Pen devices for Windows 7 and earlier operating systems.

Windows Pointer Device Data Delivery Protocol

Article • 01/18/2019

This section provides information about Windows 8 devices that support the pen, or touch, functionality.

In this section

- Design Guide

Windows Pointer Device Design Guide

Article • 01/18/2019

This section includes the following topics:

Windows Pointer Device Overview

Required Descriptors

Supporting Usages in Digitizer Report Descriptors

Button State Transitions

Multi-touch Devices

Rules for Global Items

Touch and Pen Support

Additional Resources

Windows Pointer Device Overview

Article • 11/05/2020

In Windows 8, a *Windows pointer device* refers to devices that support the pen, or touch, functionality. In the context of a Windows Pointer Device, a pen is a single contact point active stylus input, also known as tablet-pen that supports hovering. Touch functionality refers to a single finger contact point or two or more concurrent finger contacts.

Windows Pointer Devices

Windows pointer devices are expected to use the human interface device (HID) protocol in order to communicate with the host. The following documents include information about the protocol:

- Device Class Definition for Human Interface Devices (HID) Version 1.11
- HID Usage Tables
- HID Over I2C Protocol Specification Version 1.0

Because Windows 8 includes an HID class driver and corresponding HID I2C and HID USB miniport drivers, you do not need to implement one. You only need to report the usages described in this white paper in the firmware for your pointer device. Windows will use your firmware and its own HID driver to enable touch and pointer capabilities for your device and furnish the Windows touch and pointer APIs with access to your device.

To report data for any type of integrated stylus, the input must be reported by using a pen Collection Application(CA) collection. Similarly, touch data should be reported by using a touch CA collection. External stylus devices should use the digitizer CA collection. Sample descriptors are included under Implementing Top-Level Collections in Multi-touch Devices section for multi-touch devices.

Windows Hardware Certification

As part of Windows 8 touch hardware certification, digitizers are required to appear to the Windows operating system as human interface device (HID) devices and follow the rules described in Required HID Descriptors.

Required Descriptors

Article • 11/05/2020

Required OS Descriptors

If your device uses the internal USB bus, you must enable the USB selective suspend feature for your HID device by using the USB Microsoft OS descriptor. With a properly formatted Microsoft OS extended property descriptor, the USB selective suspend feature can be enabled automatically whenever the HID device is connected. For more details about how the Microsoft OS extended property descriptor can be used to automatically enable selective suspend, see Microsoft OS descriptor.

Required HID Descriptors

A Windows pointer device must provide the following HID descriptors, attributes, and strings.

Required USB HID Descriptor

The following table shows the required USB HID descriptor. For more information, see section 6.2.1 in Device Class Definition for Human Interface Devices (HID) Version 1.11 .

Expand table

Member	Size in Bytes	Description
bLength	1	Size of the descriptor
bDescriptorType	1	Type of descriptor
bcdHID	2	HID version number
bCountryCode	1	Country code
bNumDescriptors	1	Number of descriptors
bDescriptorType	1	Descriptor type
bDescriptorLength	2	Length of the descriptor

Required I2C HID Descriptor

The following table shows the required I2C HID descriptor.

Expand table

Member	Size in Bytes	Description
bLength	2	The length of the complete HID descriptor, in bytes.
bcdVersion	2	The version number, in binary coded decimal (BCD) format.
dwReportDescLength	2	The length of the Report descriptor, in bytes.
wReportDescRegister	2	The register index containing the Report descriptor.
wInputRegister	2	The register number to read the input report, in unsigned bytes.
wOutputRegister	2	The register number to send the output, in unsigned bytes.
wVendorID	2	USB-IF assigned Vendor ID.
wDeviceID	2	Device ID.
wRevisionID	2	Firmware version number

Required Device Attributes

The following HID properties must be provided in the device attributes. The reporting of these device attributes is bus specific. Consult the HID specific guidance for your choice of bus.

Expand table

Member	Description
bSize	Size of the Device descriptor
wVendorID	Vendor Id
wProduct	Product Id
Version Number	Firmware version number

HID Strings

The following strings must be supported by devices that support Windows pointer functionality:

- Manufacturer Name
- Product String

Supporting Usages in Digitizer Report Descriptors

Article • 09/30/2021

A *Usage* is the name of a value, button, or collection in a HID report. The host makes use of the following usages when extracting data from a pen or touch device.

Member	Description	Page	ID	Device
X	X coordinate of the contact position.	Desktop	0x30	Pen, Touch
Y	Y coordinate of the contact position.	Desktop	0x31	Pen, Touch
Tip	Set if the finger or pen is on the surface of the digitizer.	Digitizer	0x42	Pen, Touch
In-range	Set when the finger or pen is detected while hovering over the digitizer or in contact with the digitizer surface.	Digitizer	0x32	Pen, Touch
Confidence	Set when contact is a finger (not a palm or any other part of the hand that should not trigger finger input).	Digitizer	0x47	Touch
Width	Width of contact.	Digitizer	0x48	Touch
Height	Height of contact.	Digitizer	0x49	Touch
Scan Time	Relative scan time.	Digitizer	0x56	Touch
Pressure	Amount of pressure the user is applying to the contact point.	Digitizer	0x30	Pen, Touch
Barrel	Set if the button on the barrel of a stylus is pressed	Digitizer	0x44	Pen
Azimuth	The counter-clockwise rotation of the cursor about the Z-axis.	Digitizer	0x3f	Pen, Touch
Invert	Set when the opposite end of the pen is hovering over the digitizer.	Digitizer	0x3c	Pen
Eraser	Set when the opposite end of the pen is on the surface of the digitizer.	Digitizer	0x45	Pen
X Tilt	The angle between the Y-Z plane and the plane containing the pointer device axis and the Y axis.	Digitizer	0x3d	Pen

Member	Description	Page	ID	Device
Y Tilt	The angle between the X-Z plane and the pointer device plane; a positive Y tilt is toward the user.	Digitizer	0x3e	Pen
Twist	Clockwise rotation of the cursor about its own axis.	Digitizer	0x41	Pen

Note Usages listed in the table above are known to Windows and are delivered to applications using the WM_POINTER message.

Note Pointer devices are free to support additional usages (including vendor-specific usages). Additional usages are not delivered to applications in WM_POINTER messages. The value of these usages can be retrieved by using the GetRawPointerDeviceData function. In order to make the usages accessible from the **GetRawPointerDeviceData** function, the usages must be located in the same report as the X and Y usages.

HID Descriptor for Digitizers

Starting with Windows 8, a touch digitizer must appear as a touch screen (page = 0x0D, usage = 0x04). A stylus digitizer must appear as an integrated pen (page = 0x0D, usage = 0x02) or an external pen (page = 0x0D, usage = 0x01). Integrated touch and pen devices are mapped to the display that they are physically connected to. External pen devices are mapped to the virtual desktop.

Required HID Usages for Digitizers

The following usages are required for all digitizers. Devices that do not support all of the required usages will not work on Windows 8:

X and **Y**
X and **Y** report the coordinates of contact. In Windows 8, a device can report two points for each contact. The first point (known as T) is considered the point that the user intended to touch while the second point (known as C) is considered the center of the contact. Devices that are capable of reporting T and C should have a usage array of two X values and two Y values. The values in the first position in the arrays are interpreted as the coordinates for T and the values in the second position are interpreted as the coordinates for C. (the report count for both usages is 2 to indicate the presence of a usage array). Devices that report C must also report the **Width** and **Height** usages. The host uses C to build the bounding rectangle around the contact. If the device only reports one X and Y pair, the host uses that pair for T and C. The sample touch descriptor includes usage arrays for both **X** and **Y**. The following extracts from the

sample descriptor illustrate the difference between a device that supports only T and a device that supports T and C.

A device that reports only T must not have a usage array for the X and Y properties (that is, the report count for each usage is 1 as indicated in the following.)

```
0x05, 0x01,                         //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,                   //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                         //       REPORT_SIZE (16)
    0x55, 0x0e,                         //       UNIT_EXPONENT (-2)
    0x65, 0x13,                         //       UNIT(Inch,EngLinear)
    0x09, 0x30,                         //       USAGE (X)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,                   //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,                   //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
```

A device that supports T and C uses usage arrays for reporting the X and Y values. The report count for both **X** and **Y** is 2.

```
0x05, 0x01,                         //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,                   //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                         //       REPORT_SIZE (16)
    0x55, 0x0e,                         //       UNIT_EXPONENT (-2)
    0x65, 0x13,                         //       UNIT(Inch,EngLinear)
    0x09, 0x30,                         //       USAGE (X)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,                   //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                         //       REPORT_COUNT (2)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,                   //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
```

Note These examples take advantage of the HID rule that global items stay the same for every main item until they are changed. This allows both **X** and **Y** usages to share just one entry for the report count.

Pen devices should use the first example for their descriptors since C isn't relevant for these devices.

The following global items must be specified for the **X** and **Y** usages:

- Logical minimum
- Logical maximum
- Physical minimum
- Physical maximum
- Unit
- Unit Exponent

The physical range for the device and the units must be accurately reported. If the information is inaccurate, the device will not work correctly. Devices must also report data within the logical range that is specified in the report descriptor. Any reported value outside this range will be considered as invalid data and the value will be changed to the nearest boundary value (logical minimum or logical maximum).

Tip
Use the **Tip** switch to indicate finger or pen contact and lift-off from the digitizer surface. There should be a main item with a report size of 1. When delivering reports, the bit position should be set when the finger or pen is in contact with the digitizer surface. Otherwise, the bit should be cleared.

Scan Time
Scan Time reports relative time in units of 100 microseconds. It represents the delta from the first frame that was reported after a device starts reporting data subsequent to a period of inactivity. The first scan time received is treated as a base time for subsequent reported times. The deltas between reported scan times should reflect the scanning frequency of the digitizer. It is important to note that unlike other usages, the host does not allow any flexibility for the unit for the scan time usage. The value is expected to roll over, as only 1 byte is allocated to the counter. The scan time value should be the same for all contacts within a frame. This requirement also applies to devices that report data using the hybrid mode

Note This usage is only required for touch devices.

In-range
If the device supports Z-axis detection, the digitizer must set the **In-range** usage in the input report when the transducer is within the region where digitizing is possible. If the device does not support Z-axis detection, the driver should not include the **In-range** usage in its report descriptor.

Earlier versions of Windows have different guidelines for how touch digitizer drivers should handle in-range reporting.

Devices that support pen and touch should support NULL states for the **X** and **Y** usages in the top level collection for the pen. When the pen is detected at a height where the X

and Y values cannot be accurately detected, the device should report NULL values for X and Y and set the **In-range** usage. NULL values simply mean values outside the specified logical range for these usages provided that the device has indicated that it supports NULL for the relevant usage. The device can then report accurate X and Y values once the stylus is close enough to the surface to make this possible. This protocol allows the host to implement palm rejection when the pen is in range.

It should be noted that the host will recognize the values outside the logical range as signifying the implementation of this protocol only if the report descriptor specifically includes the bit signifying the fact that X and Y support **NULL** states. Otherwise, values outside the logical range are simply moved to the nearest boundary value. The following extracts from report descriptors illustrate the difference between a device that supports **NULL** for X and Y and one that doesn't. It should be noted that NULL support is only needed in the pen top level collection. Touch top level collections don't need to support **NULL** for X and Y for this purpose.

Report extract with **NULL** support for X and Y:

```
0x05, 0x01,                         //     USAGE_PAGE (Generic Desktop) 42
    0x09, 0x30,                         //     USAGE (X)
44
    0x75, 0x10,                         //     REPORT_SIZE (16)
46
    0x95, 0x01,                         //     REPORT_COUNT (1)
48
    0xa4,                               //     PUSH
50
    0x55, 0x0d,                         //     UNIT_EXPONENT (-3)
51
    0x65, 0x13,                         //     UNIT (Inch,EngLinear)
53
    0x35, 0x00,                         //     PHYSICAL_MINIMUM (0)
55
    0x46, 0x3a, 0x20,                   //     PHYSICAL_MAXIMUM (8250)
57
    0x26, 0xf8, 0x52,                   //     LOGICAL_MAXIMUM (21240)
60
    0x81, 0x42,                         //     INPUT (Data,Var,Abs)
63
    0x09, 0x31,                         //     USAGE (Y)
65
    0x46, 0x2c, 0x18,                   //     PHYSICAL_MAXIMUM (6188)
67
    0x26, 0x6c, 0x3e,                   //     LOGICAL_MAXIMUM (15980)
70
    0x81, 0x42,                         //     INPUT (Data,Var,Abs)
73
```

Report extract without NULL support for X and Y:

```
0x05, 0x01,                         //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,                   //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                         //       REPORT_SIZE (16)
    0x55, 0x0e,                         //       UNIT_EXPONENT (-2)
    0x65, 0x13,                         //       UNIT(Inch,EngLinear)
    0x09, 0x30,                         //       USAGE (X)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,                   //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,                   //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
```

See the Touch and Pen Support section for a full descriptor of a pen device that supports NULL values for **X** and **Y**.

Note This usage is required for all pen devices but is optional for touch devices.

Optional HID Usages

The following usages are optional, but you should implement them if the digitizer hardware supports them. Digitizers that do not support these usages should not include them in the report descriptor.

Width and **Height**
The **Width** and **Height** usages represent the width and height of the bounding box around the touch contact. The reported values should never be 0 except when an “UP” event is being reported, in which case they should be 0.

Width and **Height** are also exposed to application developers through the WM_POINTER message.

Confidence
Confidence is a suggestion from the device about whether the touch contact was an intended or accidental touch. If you are confident that the touch is intended, then set the confidence usage to 1 (true). Your device should reject accidental touches as thoroughly as it can while the latency stays within the required range. If you are not certain that the touch is intended, and your device did not reject the touch as accidental, then set the confidence usage to 0 (false). If your device always rejects accidental touches, you do not need to include the confidence usage.

Pressure

Pressure is a measurement of the force that the finger or pen exerts against the digitizer surface. There are no restrictions on the range allowed for pressure. However, the range specified by a device will be normalized to a range of 0 – 1024 when delivering data to client applications.

Barrel

Barrel should be set when the pen barrel button is pressed. Otherwise, it should be reset. Barrel is used by Windows to modify the function of the tip for a primary action (tap, drag) or a secondary action (right tap, right drag).

Although **Pressure** and **Barrel** are optional usages, implementing them for pen digitizers is recommended. Additional value is added for these usages: pressure defines pen stroke width, making it more realistic, and the barrel switch allows right mouse button functionality when using the pen.

X Tilt

X Tilt represents the plane angle between the Y-Z plane and the plane containing the transducer axis and the Y axis.

The physical range and logical range must be specified. The physical range must be -90 to 90. The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians can also be used for the physical range. The following list shows a typical logical and physical combination.

- Logical minimum: -9000
- Logical maximum: 9000
- Unit: Degrees
- Unit Exponent: -2
- Physical minimum: -9000
- Physical maximum: 9000

Y Tilt

Y Tilt represents the plane angle between the X-Z plane and the plane containing the transducer X planes.

The physical range and logical range must be specified. The physical range must be between –90 to 90. The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians can also be used for the physical range. The following list shows a typical logical and physical combination.

- Logical minimum: 0
- Logical maximum: 18000

- Unit: Degrees
- Unit Exponent: -2
- Physical minimum: -9000
- Physical maximum: 9000

Twist
Twist specifies the clockwise rotation of the cursor around its own major axis.

The physical range and logical range must be specified. The physical range must be 0 to 360. The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians can also be used for the physical range. In this case, the logical range must be large enough to report values that are accurate to at least four decimal places. The following list shows a typical logical and physical combination.

- Logical minimum: 0
- Logical maximum: 62831
- Unit: Radians
- Unit exponent: -4
- Physical minimum: 0
- Physical maximum: 62831

Azimuth
Azimuth specifies the counter-clockwise rotation of the cursor around the Z-axis through a full circular range. The physical range and logical range must be specified. The physical range must be 0 to 360, however the device may report either the full range or half the range assuming the touch ellipsis is symmetric based on hardware capabilities. The logical range must be large enough to deliver data that is accurate to at least two decimal places. Radians can also be used for the physical range. In this case, the logical range must be large enough to report values that are accurate to at least 4 decimal places. The following list shows a typical logical and physical combination.

- Logical minimum: 0
- Logical maximum: 36000
- Unit: Degrees
- Unit Exponent: -2
- Physical minimum: 0
- Physical maximum: 36000

Note The Unit Exponent must be -2 when Unit is Degrees and -4 when Unit is Radians.

Button State Transitions

Article • 01/18/2019

Certain button state transitions are expected from devices. Devices that support hovering must include the **In-range** and **Tip** usages in their descriptor.

The transitions for these devices are listed in the following table.

Event	Button Status
Device comes in range	In-range=on; Tip=off
Device comes in contact with the surface of the digitizer	In-range=on; Tip=on
Contact is moving on the digitizer surface	In-range=on; Tip=on
Contact is lifted off of the digitizer surface	In-range=off; Tip=off
Contact goes out of range	In-range=off; Tip=off

Note The X and Y coordinates reported for the “out of range” event must match those reported for the last “in range” event before the “out of range” event was detected. For the case when the device goes “out of range” very quickly, where “up” and “out of range” are detected in the same scan, both events or packets need to be reported. One for “up”, and another for “out of range”.

Devices that do not support hovering do not need to include the **In-range** usage in their descriptor. The transitions for these devices are listed in the following table.

Event	Button Status
Device comes in contact with the surface of the digitizer	Tip=on
Contact is moving on the digitizer surface	In-range=on; Tip=on
Contact is lifted off of the digitizer surface	In-range=off; Tip=off

Note The X and Y coordinates reported when the finger is lifted off the digitizer surface must be the same as those reported for the last “move” packet detected.

The transitions for pen devices that support the Eraser and Inverted usages are given in the following table. These are the only valid states. Any other state may be rejected by Windows.

Event	Button Status

Event	Button Status
Tip end of pen is hovering	In-range=on; Tip=off; Inverted=off; Eraser=off
Tip end of pen is on the surface	In-range=on; Tip=on; Inverted=off; Eraser=off
Tip end of pen is off the surface and hovering again	In-range=on; Tip=off; Inverted=off; Eraser=off
Tip goes out of range	In-range=off; Tip=off; Inverted=off; Eraser=off
Eraser end of the pen is hovering	In-range=on; Tip=off; Inverted=on; Eraser=off
Eraser end of the pen is on the surface	In-range=on; Tip=off; Inverted=off; Eraser=on
Eraser end of the pen is off the surface and hovering again	In-range=on; Tip=off; Inverted=on; Eraser=off
Eraser goes out of range	In-range=off; Tip=off; Inverted=off; Eraser=off

Multi-touch Devices

Article • 01/18/2019

This section includes topics for multi-touch devices.

Supporting Usages in Multi-touch Digitizer Drivers

Selecting Packet Reporting Modes in Multi-touch Devices

Implementing Top-Level Collections in Multi-touch Devices

Supporting Usages in Multi-touch Digitizers

Article • 10/25/2024

In the context of a Windows Pointer Device, multi-touch refers to support for two or more contact points. The required and optional usages for a multi-touch digitizer device are described below.

Required HID Usages for Multi-touch Digitizers

The report descriptor for a multi-touch digitizer must specify that the device is a HID touch screen (page = 0x0D and usage = 0x04).

In addition to the existing HID touch usages, multi-touch digitizers must implement the following usages.

- Contact identifier
- Contact count maximum
- Scan Time

Optional HID Usages

The following usages are optional, but multi-touch digitizers should repoirt them if the digitizer hardware supports them.

- Confidence
- Pressure
- Azimuth
- In-range (optional for touch, required for pen)
- Width and Height

HID Usages for Multi-touch Digitizers

The HID standard defines the following usages for multi-touch input from digitizers.

Expand table

Name	Description	CA Usage	Page	Type	ID
Contact Identifier	Contact identifier	Touch	Digitizer	Dynamic Value (DV)	0x51
Contact Count	Actual contact count	Touch	Digitizer	Dynamic Value (DV)	0x54
Contact count maximum	Maximum number of contacts supported	Touch	Digitizer	Dynamic Value (DV)	0x55

Contact identifier Specifies the identifier of the current contact. An identifier must remain constant while the contact is detected by the device. Each separate concurrent contact must have a unique identifier. Identifiers can be reused if a contact is no longer detected. If the device supports "in-air" packets (the contact is hovering above the surface), the identifier must persist from the time that the contact is detected until the time that it goes out of range.

Contact count Specifies the number of valid contacts in the current packet. Drivers that use parallel or Hybrid mode should include this usage. A device that cannot provide this value must use **NULL** for all values in the first position that do not contain valid contact information. However, **NULL** is an option only for Parallel mode devices. Devices should not use a combination of Contact count and **NULL** for reporting the actual count. Either one or the other should be used.

Contact count maximum Specifies the total number of contacts that a multi-touch device supports. This usage must be included in the multi-touch top-level collection and not in any child collection. This usage must be present in a feature report in the touch top-level collection. While reporting data, a device must not report more contacts than the contact count maximum. A device without the Contact count maximum in the descriptor will be considered to be a single-touch device. While reporting data, a device must not report more contacts than the Contact Count maximum. If an input frame contains more contacts than the maximum, the host will discard the entire frame, including the data for any previously reported contacts. For the best user experience, the device should suppress the reporting of any new contacts detected after the maximum number has been reached for the lifetime of the new contact, even if any existing contacts depart. Existing contacts can continue to be reported, and if any depart, newly arriving contacts can take their place.

Selecting Packet Reporting Modes in Multi-touch Devices

Article • 01/18/2019

Starting with Windows 8, the reporting of multi-touch data to the system is supported in two ways: Parallel mode or Hybrid mode. The vendor-supplied HID report descriptor differs, depending on the mode that was selected.

Note Serial mode is selected by legacy devices only. Any new devices using serial mode will fail certification starting with Windows 8.

Parallel Mode

In Parallel mode, devices report all contact information in a single packet. Each physical contact is represented by a logical collection that is embedded in the top-level collection. This logical collection contains all the usages that the device supports for each contact (for example, **X**, **Y**, and **Pressure**). When taking advantage of Parallel mode, each of the logical collections must be identical. Because the device generally reports fewer contacts than the maximum, the number of contacts that are reported in a parallel packet should be communicated either in the Contact count usage or by setting NULL values for all invalid contacts in a packet.

For example, consider a device that supports three contacts. If the user has only two fingers on the digitizer, the parallel packet has only two valid contact data in a report that can carry data for three contacts. In this case, the Contact count should be set to 2 so that the client application knows that any information about more than two contacts is not valid.

Alternatively, the device can set the values of the contact usages beyond the second entry to **NULL** values. Considering the overhead for supporting and reporting **NULL** values, using the actual count is encouraged.

A disadvantage of reporting multiple contacts in one report is that space is wasted per report every time that there are fewer contacts than the maximum number of contacts possible. Devices can use the Hybrid mode to reduce this inefficiency.

Hybrid Mode

In Hybrid mode, the number of contacts that can be reported in one report is less than the maximum number of contacts that the device supports. For example, a device that

supports a maximum of 48 concurrent physical contacts can set up its top-level collection to deliver a maximum of 12 contacts in one report. If 48 contact points are present, the device can break these down into 4 serial reports that deliver 12 contacts each.

When a device delivers data in this manner, the Contact count usage value in the first report should reflect the total number of contacts that are being delivered in the hybrid reports. The other serial reports should have a contact count of 0. Using the previous example, the Contact count usage in the first report has a value of 48, whereas the latter three reports have a contact usage count of 0. Due to this data delivery protocol, the actual count usage must be present in the input report of the report descriptor for the device.

Null Values

Null values should be specified as outlined in the HID specification. The **NULL** bit must be set on all main items in the report descriptor. Be aware that a device can use either the Contact count usage or **NULL** values to notify the host of the actual number of valid contacts in a packet.

Contacts in a Report

When sending data in hybrid or parallel mode, a contact that is delivered in one report must be delivered in all subsequent reports until it is lifted off the screen. If time is needed to adequate determine if the contact was lifted off the surface, the device must report the last known position of the contact and then deliver the “UP” state of the contact in a subsequent report. Devices should not send a report without the information for that contact while trying to determine its current state.

Implementing Top-Level Collections in Multi-touch Devices

Article • 01/18/2019

The report descriptor for a multiple input device must include at least one top-level collection for the primary device and a separate top-level collection for the mouse.

Report descriptors for touch devices should use Finger (0x22) CL (Collection Logical) to group the data and control usages in the top-level collections, whereas the Stylus (0x20) CL should be used to group pen-related control and data usages.

The following report descriptor samples demonstrate the collection and usage settings for device type and mode.

- Mouse Collection Report Descriptor
- Two Finger Parallel/Hybrid Mode Report Descriptor
- Single Finger Hybrid Mode Report Descriptor

Mouse Collection Report Descriptor

Article • 01/24/2019

The mouse report descriptor sample shows the pointer device usage in a physical collection.

```
//
    // Dummy mouse collection starts here
    //
    0x05, 0x01,                         // USAGE_PAGE (Generic Desktop)
0
    0x09, 0x02,                         // USAGE (Mouse)
2
    0xa1, 0x01,                         // COLLECTION (Application)
4
    0x85, REPORTID_MOUSE,               //   REPORT_ID (Mouse)
6
    0x09, 0x01,                         //   USAGE (Pointer)
8
    0xa1, 0x00,                         //   COLLECTION (Physical)
10
    0x05, 0x09,                         //     USAGE_PAGE (Button)
12
    0x19, 0x01,                         //     USAGE_MINIMUM (Button 1)
14
    0x29, 0x02,                         //     USAGE_MAXIMUM (Button 2)
16
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
18
    0x25, 0x01,                         //     LOGICAL_MAXIMUM (1)
20
    0x75, 0x01,                         //     REPORT_SIZE (1)
22
    0x95, 0x02,                         //     REPORT_COUNT (2)
24
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
26
    0x95, 0x06,                         //     REPORT_COUNT (6)
28
    0x81, 0x03,                         //     INPUT (Cnst,Var,Abs)
30
    0x05, 0x01,                         //     USAGE_PAGE (Generic Desktop)
32
    0x09, 0x30,                         //     USAGE (X)
34
    0x09, 0x31,                         //     USAGE (Y)
36
    0x15, 0x81,                         //     LOGICAL_MINIMUM (-127)
38
```

```
    0x25, 0x7f,                         //     LOGICAL_MAXIMUM (127)
40
    0x75, 0x08,                         //     REPORT_SIZE (8)
42
    0x95, 0x02,                         //     REPORT_COUNT (2)
44
    0x81, 0x06,                         //     INPUT (Data,Var,Rel)
46
    0xc0,                               //   END_COLLECTION
48
    0xc0                                // END_COLLECTION
49/50
```

Two Finger Parallel/Hybrid Mode Report Descriptor

Article • 01/24/2019

The Two Finger Parallel/Hybrid Mode Multi-touch Device sample report descriptor contains two logical collections (one for each finger). Be aware that this descriptor can still support more than two concurrent contacts by using Hybrid mode.

```
    0x05, 0x0d,                    // USAGE_PAGE (Digitizers)
    0x09, 0x04,                    // USAGE (Touch Screen)
    0xa1, 0x01,                    // COLLECTION (Application)
    0x85, 0x01,                    //   REPORT_ID (Touch)
    0x09, 0x22,                    //   USAGE (Finger)
    0xa1, 0x02,                    //     COLLECTION (Logical)
    0x09, 0x42,                    //       USAGE (Tip Switch)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //       REPORT_SIZE (1)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x07,                    //       REPORT_COUNT (7)
    0x81, 0x03,                    //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                    //       REPORT_SIZE (8)
    0x09, 0x51,                    //       USAGE (Contact Identifier)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x01,                    //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,              //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                    //       REPORT_SIZE (16)
    0x55, 0x0e,                    //       UNIT_EXPONENT (-2)
    0x65, 0x13,                    //       UNIT(Inch,EngLinear)
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,              //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                    //       REPORT_COUNT (2)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,              //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                    //       USAGE (Y)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x0d,                    //       USAGE_PAGE (Digitizers)
    0x09, 0x48,                    //       USAGE (Width)
    0x09, 0x49,                    //       USAGE (Height)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x55, 0x0C,                    //       UNIT_EXPONENT (-4)
    0x65, 0x12,                    //       UNIT (Radians,SIROtation)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x47, 0x6f, 0xf5, 0x00, 0x00,  //       PHYSICAL_MAXIMUM (62831)
```

```
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x27, 0x6f, 0xf5, 0x00, 0x00,       //       LOGICAL_MAXIMUM (62831)
    0x09, 0x3f,                         //       USAGE
(Azimuth[Orientation])
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0xc0,                               //     END_COLLECTION
    0x09, 0x22,                         //   USAGE (Finger)
    0xa1, 0x02,                         //     COLLECTION (Logical)
    0x09, 0x42,                         //       USAGE (Tip Switch)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                         //       REPORT_SIZE (1)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x95, 0x07,                         //       REPORT_COUNT (7)
    0x81, 0x03,                         //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                         //       REPORT_SIZE (8)
    0x09, 0x51,                         //       USAGE (Contact Identifier)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x05, 0x01,                         //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,                   //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                         //       REPORT_SIZE (16)
    0x55, 0x0e,                         //       UNIT_EXPONENT (-2)
    0x65, 0x13,                         //       UNIT(Inch,EngLinear)
    0x09, 0x30,                         //       USAGE (X)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,                   //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                         //       REPORT_COUNT (2)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,                   //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                         //       USAGE (Y)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x05, 0x0d,                         //       USAGE_PAGE (Digitizers)
    0x09, 0x48,                         //       USAGE (Width)
    0x09, 0x49,                         //       USAGE (Height)
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0x95, 0x01,                         //       REPORT_COUNT (1)
    0x55, 0x0C,                         //       UNIT_EXPONENT (-4)
    0x65, 0x12,                         //       UNIT (Radians,SIROtation)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x47, 0x6f, 0xf5, 0x00, 0x00,       //       PHYSICAL_MAXIMUM (62831)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x27, 0x6f, 0xf5, 0x00, 0x00,       //       LOGICAL_MAXIMUM (62831)
    0x09, 0x3f,                         //       USAGE
(Azimuth[Orientation])
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0xc0,                               //     END_COLLECTION
    0x05, 0x0d,                         //   USAGE_PAGE (Digitizers)
    0x55, 0x0C,                         //     UNIT_EXPONENT (-4)
    0x66, 0x01, 0x10,                   //     UNIT (Seconds)
    0x47, 0xff, 0xff, 0x00, 0x00,       //       PHYSICAL_MAXIMUM (65535)
    0x27, 0xff, 0xff, 0x00, 0x00,       //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                         //   REPORT_SIZE (16)
    0x95, 0x01,                         //   REPORT_COUNT (1)
```

```
    0x09, 0x56,                         //   USAGE (Scan Time)
    0x81, 0x02,                         //   INPUT (Data,Var,Abs)
    0x09, 0x54,                         //   USAGE (Contact count)
    0x25, 0x7f,                         //   LOGICAL_MAXIMUM (127)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x75, 0x08,                         //   REPORT_SIZE (8)
    0x81, 0x02,                         //   INPUT (Data,Var,Abs)
    0x85, REPORTID_MAX_COUNT,           //   REPORT_ID (Feature)
    0x09, 0x55,                         //   USAGE(Contact Count Maximum)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x25, 0x02,                         //   LOGICAL_MAXIMUM (2)
    0xb1, 0x02,                         //   FEATURE (Data,Var,Abs)
    0x85, 0x44,                         //   REPORT_ID (Feature)
    0x06, 0x00, 0xff,                   //   USAGE_PAGE (Vendor Defined)
    0x09, 0xC5,                         //   USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //   LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //   REPORT_SIZE (8)
    0x96, 0x00,  0x01,                  //   REPORT_COUNT (0x100 (256))
    0xb1, 0x02,                         //   FEATURE (Data,Var,Abs)
    0xc0,                               // END_COLLECTION
```

Single Finger Hybrid Mode Report Descriptor

Article • 01/09/2024

The Single Finger Hybrid Mode Multi-touch Devices sample report descriptor contains a single logical collection. For many categories of devices, end users may operate with a single finger in most scenarios. For this type of device, this mode is the most optimal.

Note HID I2C multi-touch devices are required to follow this mode.

```
0x05, 0x0d,                    // USAGE_PAGE (Digitizers)
    0x09, 0x04,                    // USAGE (Touch Screen)
    0xa1, 0x01,                    // COLLECTION (Application)
    0x85, 0x01,                    //   REPORT_ID (Touch)
    0x09, 0x22,                    //   USAGE (Finger)
    0xa1, 0x02,                    //     COLLECTION (Logical)
    0x09, 0x42,                    //       USAGE (Tip Switch)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //       REPORT_SIZE (1)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x07,                    //       REPORT_COUNT (7)
    0x81, 0x03,                    //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                    //       REPORT_SIZE (8)
    0x09, 0x51,                    //       USAGE (Contact Identifier)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x01,                    //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x0f,              //       LOGICAL_MAXIMUM (4095)
    0x75, 0x10,                    //       REPORT_SIZE (16)
    0x55, 0x0e,                    //       UNIT_EXPONENT (-2)
    0x65, 0x13,                    //       UNIT(Inch,EngLinear)
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0xb5, 0x04,              //       PHYSICAL_MAXIMUM (1205)
    0x95, 0x02,                    //       REPORT_COUNT (2)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x46, 0x8a, 0x03,              //       PHYSICAL_MAXIMUM (906)
    0x09, 0x31,                    //       USAGE (Y)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x0d,                    //       USAGE_PAGE (Digitizers)
    0x09, 0x48,                    //       USAGE (Width)
    0x09, 0x49,                    //       USAGE (Height)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x55, 0x0C,                    //       UNIT_EXPONENT (-4)
```

```
    0x65, 0x12,                         //       UNIT (Radians,SIROtation)
    0x35, 0x00,                         //       PHYSICAL_MINIMUM (0)
    0x47, 0x6f, 0xf5, 0x00, 0x00,       //       PHYSICAL_MAXIMUM (62831)
    0x15, 0x00,                         //       LOGICAL_MINIMUM (0)
    0x27, 0x6f, 0xf5, 0x00, 0x00,       //       LOGICAL_MAXIMUM (62831)
    0x09, 0x3f,                         //       USAGE
 (Azimuth[Orientation])
    0x81, 0x02,                         //       INPUT (Data,Var,Abs)
    0xc0,                               //     END_COLLECTION
    0x05, 0x0d,                         //   USAGE_PAGE (Digitizers)
    0x55, 0x0C,                         //   UNIT_EXPONENT (-4)
    0x66, 0x01, 0x10,                   //   UNIT (Seconds)
    0x47, 0xff, 0xff, 0x00, 0x00,       //   PHYSICAL_MAXIMUM (65535)
    0x27, 0xff, 0xff, 0x00, 0x00,       //   LOGICAL_MAXIMUM (65535)
    0x75, 0x10,                         //   REPORT_SIZE (16)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x09, 0x56,                         //   USAGE (Scan Time)
    0x81, 0x02,                         //   INPUT (Data,Var,Abs)
    0x09, 0x54,                         //   USAGE (Contact count)
    0x25, 0x7f,                         //   LOGICAL_MAXIMUM (127)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x75, 0x08,                         //   REPORT_SIZE (8)
    0x81, 0x02,                         //   INPUT (Data,Var,Abs)
    0x85, REPORTID_MAX_COUNT,           //   REPORT_ID (Feature)
    0x09, 0x55,                         //   USAGE(Contact Count Maximum)
    0x95, 0x01,                         //   REPORT_COUNT (1)
    0x25, 0x02,                         //   LOGICAL_MAXIMUM (2)
    0xb1, 0x02,                         //   FEATURE (Data,Var,Abs)
    0x85, 0x44,                         //   REPORT_ID (Feature)
    0x06, 0x00, 0xff,                   //   USAGE_PAGE (Vendor Defined)
    0x09, 0xC5,                         //   USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //   LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //   LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //   REPORT_SIZE (8)
    0x96, 0x00, 0x01,                   //   REPORT_COUNT (0x100 (256))
    0xb1, 0x02,                         //   FEATURE (Data,Var,Abs)
    0xc0,                               // END_COLLECTION
```

Note the REPORTID_MAX_COUNT and REPORTID_MOUSE referenced in the above descriptors don't signify any particular value. Report identifiers are optional and should be assigned according to the needs of every report descriptor. Please see the Device Class Definition for Human Interface Devices (HID) Version 1.11 for more information about report identifiers.

The following example shows 2 frames of 5 reports each for single-finger hybrid device representing 5 counts.

Expand table

X	Y	Tip	Scan Time	Contact ID	Contact Count
517	350	TRUE	3472	0	5
706	217	TRUE	3472	1	0
510	640	TRUE	3472	2	0
1113	317	TRUE	3472	3	0
866	218	TRUE	3472	4	0
517	350	TRUE	3561	0	5
706	217	TRUE	3561	1	0
510	640	TRUE	3561	2	0
1113	317	TRUE	3561	3	0
866	218	TRUE	3561	4	0

Configuration Top Level Collection

Article • 01/18/2019

In previous versions of windows, some devices used a configuration top level collection to allow their devices to adapt to the capabilities of the host operating system. This functionality is available in Windows 8 for backwards compatibility but it is strongly recommended not to use the configuration collection when designing a new digitizer for Windows. This feature is deprecated in Windows 8 and will be removed in a future version of Windows, and is not available in Windows RT.

Rules for Global Items

Article • 01/18/2019

The following rules are established for reporting global items in descriptors.

- A logical range must be provided for all usages.
- The minimum physical size of a digitizer is one square inch.
- The recommended (and minimum) time stamp usage report data size is 16 bits.
- Reported values must always be within the logical range. The only exception is when the usage includes support for **NULL** values.

Touch and Pen Support

Article • 01/24/2019

If your device includes a digitizer that provides both touch and pen functionality, you must report touch and pen collections separately. If your digitizer might be installed on a slate, tablet PC or other computer that has no mouse, you should also report a mouse collection.

Reporting Null Values

To facilitate palm rejection, the pen device should report data at a vertical distance higher than it normally would. It is highly recommended that this distance be 50 mm (or as close to this as possible), as this produces a better user experience. This will allow the host to handle touch input differently when it's aware that the pen is in range. Since most devices that are able to detect the pen at a higher vertical distance won't be able to accurately detect the coordinates of the pen, NULL coordinates should be reported by the device. The report descriptor below shows how NULL values can be supported for X and Y. When delivering NULL data, the device simply needs to set values that are outside the specified logical range for X and Y. Both values must be NULL for the host to recognize this reporting mode.

```
0x05, 0x0d,                         // USAGE_PAGE (Digitizers)          0
    0x09, 0x02,                         // USAGE (Pen)
2
    0xa1, 0x01,                         // COLLECTION (Application)
4
    0x85, 0x02,                         //   REPORT_ID (Pen)
6
    0x09, 0x20,                         //   USAGE (Stylus)
8
    0xa1, 0x00,                         //   COLLECTION (Physical)
10
    0x09, 0x42,                         //     USAGE (Tip Switch)
12
    0x09, 0x44,                         //     USAGE (Barrel Switch)
14
    0x09, 0x3c,                         //     USAGE (Invert)
16
    0x09, 0x45,                         //     USAGE (Eraser Switch)
18
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
20
    0x25, 0x01,                         //     LOGICAL_MAXIMUM (1)
22
```

```
    0x75, 0x01,                    //     REPORT_SIZE (1)
24
    0x95, 0x04,                    //     REPORT_COUNT (4)
26
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
28
    0x95, 0x01,                    //     REPORT_COUNT (1)
30
    0x81, 0x03,                    //     INPUT (Cnst,Var,Abs)
32
    0x09, 0x32,                    //     USAGE (In Range)
34
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
36
    0x95, 0x02,                    //     REPORT_COUNT (2)
38
    0x81, 0x03,                    //     INPUT (Cnst,Var,Abs)
40
    0x05, 0x01,                    //     USAGE_PAGE (Generic Desktop)
42
    0x09, 0x30,                    //     USAGE (X)
44
    0x75, 0x10,                    //     REPORT_SIZE (16)
46
    0x95, 0x01,                    //     REPORT_COUNT (1)
48
    0xa4,                          //     PUSH
50
    0x55, 0x0d,                    //     UNIT_EXPONENT (-3)
51
    0x65, 0x13,                    //     UNIT (Inch,EngLinear)
53
    0x35, 0x00,                    //     PHYSICAL_MINIMUM (0)
55
    0x46, 0x3a, 0x20,              //     PHYSICAL_MAXIMUM (8250)
57
    0x26, 0xf8, 0x52,              //     LOGICAL_MAXIMUM (21240)
60
    0x81, 0x42,                    //     INPUT (Data,Var,Abs)
63
    0x09, 0x31,                    //     USAGE (Y)
65
    0x46, 0x2c, 0x18,              //     PHYSICAL_MAXIMUM (6188)
67
    0x26, 0x6c, 0x3e,              //     LOGICAL_MAXIMUM (15980)
70
    0x81, 0x42,                    //     INPUT (Data,Var,Abs)
73
    0xb4,                          //     POP
75
    0x05, 0x0d,                    //     USAGE_PAGE (Digitizers)
76
    0x09, 0x30,                    //     USAGE (Tip Pressure)
78
    0x26, 0xff, 0x00,              //     LOGICAL_MAXIMUM (255)
```

```
80
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
83
    0x75, 0x08,                         //     REPORT_SIZE (8)
85
    0x09, 0x3d,                         //     USAGE (X Tilt)
87
    0x65, 0x14,                         //     UNIT (Degrees,EngRotation)
    0x55, 0x0e,                         //     UNIT_EXPONENT (-2)
    0x36, 0xd8, 0xdc,                   //     PHYSICAL_MINIMUM (-9000)
    0x46, 0x28, 0x23,                   //     PHYSICAL_MAXIMUM (9000)
    0x16, 0xd8, 0xdc,                   //     LOGICAL_MINIMUM (-9000)
    0x26, 0x28, 0x23,                   //     LOGICAL_MAXIMUM (9000)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x09, 0x3e,                         //     USAGE (Y Tilt)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x55, 0x0e,                         //     UNIT_EXPONENT (-2)
    0x35, 0x00,                         //     PHYSICAL_MINIMUM (0)
    0x47, 0xa0, 0x8c, 0x00, 0x00,       //     PHYSICAL_MAXIMUM (36000)
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
    0x27, 0xa0, 0x8c, 0x00, 0x00,       //     LOGICAL_MAXIMUM (36000)
    0x09, 0x41,                         //     USAGE (Twist)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0xc0,                               //   END_COLLECTION
    0xc0,                               // END_COLLECTION
```

Touch Hardware Quality Assurance

When a device passes the logo requirements, Microsoft will issue a cryptographically signed binary blob to the device's manufacturer. The manufacturer will place this blob into the device's firmware prior to production. Within Windows, when a touch device attempts to connect, the signature will be verified.

The blob itself will consist of 256 bytes of binary data, and should be reported as illustrated by the highlighted lines in the HID descriptor below. Device manufacturers should be sure that prior to issuance of the signed binary blob from Microsoft that the sample blob provided below is presented to Windows instead.

```
0x06, 0x00, 0xff,                   //     USAGE_PAGE (Vendor Defined)
    0x09, 0xC5,                         //     USAGE (Vendor Usage 0xC5)
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
    0x26, 0xff, 0x00,                   //     LOGICAL_MAXIMUM (0xff)
    0x75, 0x08,                         //     REPORT_SIZE (8)
    0x96, 0x00, 0x01,                   //     REPORT_COUNT (0x100 (256))
    0xb1, 0x02,                         //     FEATURE (Data,Var,Abs)
```

The following is the sample blob in clear text.

```
0xfc, 0x28, 0xfe, 0x84, 0x40, 0xcb, 0x9a, 0x87, 0x0d, 0xbe, 0x57, 0x3c,
0xb6, 0x70, 0x09, 0x88,
0x07, 0x97, 0x2d, 0x2b, 0xe3, 0x38, 0x34, 0xb6, 0x6c, 0xed, 0xb0, 0xf7,
0xe5, 0x9c, 0xf6, 0xc2,
0x2e, 0x84, 0x1b, 0xe8, 0xb4, 0x51, 0x78, 0x43, 0x1f, 0x28, 0x4b, 0x7c,
0x2d, 0x53, 0xaf, 0xfc,
0x47, 0x70, 0x1b, 0x59, 0x6f, 0x74, 0x43, 0xc4, 0xf3, 0x47, 0x18, 0x53,
0x1a, 0xa2, 0xa1, 0x71,
0xc7, 0x95, 0x0e, 0x31, 0x55, 0x21, 0xd3, 0xb5, 0x1e, 0xe9, 0x0c, 0xba,
0xec, 0xb8, 0x89, 0x19,
0x3e, 0xb3, 0xaf, 0x75, 0x81, 0x9d, 0x53, 0xb9, 0x41, 0x57, 0xf4, 0x6d,
0x39, 0x25, 0x29, 0x7c,
0x87, 0xd9, 0xb4, 0x98, 0x45, 0x7d, 0xa7, 0x26, 0x9c, 0x65, 0x3b, 0x85,
0x68, 0x89, 0xd7, 0x3b,
0xbd, 0xff, 0x14, 0x67, 0xf2, 0x2b, 0xf0, 0x2a, 0x41, 0x54, 0xf0, 0xfd,
0x2c, 0x66, 0x7c, 0xf8,
0xc0, 0x8f, 0x33, 0x13, 0x03, 0xf1, 0xd3, 0xc1, 0x0b, 0x89, 0xd9, 0x1b,
0x62, 0xcd, 0x51, 0xb7,
0x80, 0xb8, 0xaf, 0x3a, 0x10, 0xc1, 0x8a, 0x5b, 0xe8, 0x8a, 0x56, 0xf0,
0x8c, 0xaa, 0xfa, 0x35,
0xe9, 0x42, 0xc4, 0xd8, 0x55, 0xc3, 0x38, 0xcc, 0x2b, 0x53, 0x5c, 0x69,
0x52, 0xd5, 0xc8, 0x73,
0x02, 0x38, 0x7c, 0x73, 0xb6, 0x41, 0xe7, 0xff, 0x05, 0xd8, 0x2b, 0x79,
0x9a, 0xe2, 0x34, 0x60,
0x8f, 0xa3, 0x32, 0x1f, 0x09, 0x78, 0x62, 0xbc, 0x80, 0xe3, 0x0f, 0xbd,
0x65, 0x20, 0x08, 0x13,
0xc1, 0xe2, 0xee, 0x53, 0x2d, 0x86, 0x7e, 0xa7, 0x5a, 0xc5, 0xd3, 0x7d,
0x98, 0xbe, 0x31, 0x48,
0x1f, 0xfb, 0xda, 0xaf, 0xa2, 0xa8, 0x6a, 0x89, 0xd6, 0xbf, 0xf2, 0xd3,
0x32, 0x2a, 0x9a, 0xe4,
0xcf, 0x17, 0xb7, 0xb8, 0xf4, 0xe1, 0x33, 0x08, 0x24, 0x8b, 0xc4, 0x43,
0xa5, 0xe5, 0x24, 0xc2
```

A complete report descriptor example with the feature report containing the certification blob is shown in Single Finger Hybrid Mode Report Descriptor.

Legacy Windows Touch Drivers (Windows 7)

Article • 01/18/2019

This section presents information about how to implement support for Windows Touch devices in Windows 7 and earlier operating systems.

In this section

- Design Guide
- Windows Touch Samples

Windows Touch Design Guide

Article • 11/05/2020

This section includes the following topics:

Windows Touch Overview

Supporting Usages in Touch Digitizer Drivers

Supporting Usages in Multi-touch Digitizer Drivers

Selecting Packet Reporting Modes in Multi-touch Drivers

Implementing Top-Level Collections in Multi-touch Drivers

Using Report Descriptors to Support Capability Discovery

Additional Resources

For information about HID minidrivers for pen devices, see Requirements on Tablet PC HID Minidrivers.

Windows Touch Overview (Windows 7)

Article • 11/05/2020

Windows Touch is the name for the touch and multi-touch functionality in the Windows 7 operating system. In the context of Windows Touch, *touch* refers to support of a single physical contact point, whereas *multi-touch* refers to support for two or more concurrent physical contacts.

Choosing to Provide a Driver

In Windows 7, vendors who support Windows Touch might be required to provide a driver. If your digitizer device supports HID in firmware, you are not required to provide a driver. If your device does not support HID in firmware, you must include a driver that simulates HID support.

We recommend that touch devices be USB HID devices, and that vendors do not supply a driver. In this scenario, the report descriptor and related information is provided in firmware.

Whether you provide a driver or not, you must support selective suspend in your INF file. For more information about how to support selective suspend, see Enabling USB Selective Suspend for HID Devices.

A vendor-supplied driver should limit the processing that it does to avoid slower system performance and shorter battery life in mobile scenarios. Touch devices should process as much as they can in firmware to provide an optimal user experience.

Driver Model

If you provide a driver, we recommend that you write a KMDF-based lower filter driver. Your driver should provide the same functionality as a HID minidriver, but register as a filter driver under a minimal WDM driver (also known as a shim driver). A shim driver is necessary because KMDF 1.9 does not natively support HID minidrivers. In Windows 7 and later versions of Windows, you can use the system-supplied Mshidkmdf.sys driver as a shim.

Mshidkmdf.sys is not system-supplied in earlier versions of Windows. If you are supporting versions of Windows earlier than Windows 7, you can build the shim driver yourself. The MSDN Code Gallery contains the source code for this driver in the hidmapper subdirectory of the sample package. For more information about how to build the shim driver, see the HIDUSBFX2 sample readme in the MSDN Code Gallery.

WDM is not recommended for any natural input driver, including drivers that support Windows Touch.

Samples

The WacomKMDF and EloMT sample drivers show how to pair Mshidkmdf.sys with a vendor-supplied lower filter driver.

EloMT is a KMDF-based sample digitizer driver that provides multi-touch support. WacomKMDF is a KMDF-based sample pen driver.

Both samples use Mshidkmdf.sys as the nominal HID minidriver while the vendor-supplied KMDF driver is a lower filter driver under Mshidkmdf.sys. Mshidkmdf.sys forwards IRPs to the vendor-supplied lower filter driver.

The EloMT sample includes all the functionality that is required for a multi-touch driver to run on Windows 7. The Elotouch driver can work as both a multi-touch and a mouse driver.

Supporting Usages in Touch Digitizer Drivers (Windows 7)

Article • 11/05/2020

In the context of Windows Touch, *touch* refers to support of a single trackable contact point. This topic outlines required and optional HID usages for a touch digitizer driver. If your digitizer device supports more than one contact point, see Supporting Usages in Multi-touch Digitizer Drivers.

Usage identifier values are defined in the HID Usage Tables.

Required HID Usages

For Windows 7, a touch digitizer must appear through HID as a touch screen (page 0x0D, usage 0x04).

The following usages are required:

- X (page 0x01, usage 0x30) and Y (page 0x01, usage 0x31)

  Report x and y position location.

- Tip switch (page 0x0D, usage 0x42)

  Use the tip switch to indicate finger contact and liftoff from the digitizer surface, similar to how a pen reports contact with the digitizer.

- In-range (page 0x0D, usage 0x32)

  If the device supports z-axis detection, it reports in-range when the transducer is within the region where digitizing is possible. If the device does not support z-axis detection, the driver should set in-range and tip switch when a finger comes in contact with the digitizer.

  Versions of Windows earlier than Windows 7 have different guidelines for how touch digitizer drivers should handle in-range reporting.

Optional HID Usages

The following usages are optional, but you should implement them if the digitizer hardware supports them. These usages were added to the USB HID Usage Tables in the Windows Vista timeframe.

- Confidence (page 0x0D, usage 0x47)

  Confidence is a suggestion from the device about whether the touch contact was an intended or accidental touch. Your device should reject accidental touches as thoroughly as it can and report that information by using the confidence usage. The operating system uses confidence to help improve accidental touch rejection. In addition to the confidence value, Windows 7 applies additional heuristics to the touch input stream to improve accidental touch rejection. If your device does not report confidence, it is completely up to your device to provide accidental touch rejection.

- Width and height (page 0x0D, usages 0x48 and 0x49)

  The width and height usages represent the width and height of the touch contact. Width and height are also exposed to application developers through the Windows Touch platform.

- Pressure (page 0x0D, usage 0x30)

  Pressure is a measure of the force that the finger exerts against the digitizer surface.

For a sample touch descriptor, see Sample Report Descriptor for a Touch Digitizer Device.

Sample Report Descriptor for a Touch Digitizer Device (Windows 7)

Article • 01/24/2019

The following example shows a sample report descriptor for a touch digitizer device that supports a single contact point:

```
    0x05, 0x0d,                         // USAGE_PAGE (Digitizers)
    0x09, 0x04,                         // USAGE (Touch Screen)
    0xa1, 0x01,                         // COLLECTION (Application)
    0x85, REPORTID_TOUCH,               //   REPORT_ID (Touch)
    0x09, 0x20,                         //   USAGE (Stylus)
    0xa1, 0x00,                         //   COLLECTION (Physical)
    0x09, 0x42,                         //     USAGE (Tip Switch)
    0x15, 0x00,                         //     LOGICAL_MINIMUM (0)
    0x25, 0x01,                         //     LOGICAL_MAXIMUM (1)
    0x75, 0x01,                         //     REPORT_SIZE (1)
    0x95, 0x01,                         //     REPORT_COUNT (1)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x95, 0x03,                         //     REPORT_COUNT (3)
    0x81, 0x03,                         //     INPUT (Cnst,Ary,Abs)
    0x09, 0x32,                         //     USAGE (In Range)
    0x09, 0x47,                         //     USAGE (Confidence)
    0x95, 0x02,                         //     REPORT_COUNT (2)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x95, 0x0a,                         //     REPORT_COUNT (10)
    0x81, 0x03,                         //     INPUT (Cnst,Ary,Abs)
    0x05, 0x01,                         //     USAGE_PAGE (Generic Desktop)
    0x26, 0xff, 0x7f,                   //     LOGICAL_MAXIMUM (32767)
    0x75, 0x10,                         //     REPORT_SIZE (16)
    0x95, 0x01,                         //     REPORT_COUNT (1)
    0xa4,                               //     PUSH
    0x55, 0x0d,                         //     UNIT_EXPONENT (-3)
    0x65, 0x00,                         //     UNIT (None)
    0x09, 0x30,                         //     USAGE (X)
    0x35, 0x00,                         //     PHYSICAL_MINIMUM (0)
    0x46, 0x00, 0x00,                   //     PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x09, 0x31,                         //     USAGE (Y)
    0x46, 0x00, 0x00,                   //     PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0xb4,                               //     POP
    0x05, 0x0d,                         //     USAGE PAGE (Digitizers)
    0x09, 0x48,                         //     USAGE (Width)
    0x09, 0x49,                         //     USAGE (Height)
    0x95, 0x02,                         //     REPORT_COUNT (2)
    0x81, 0x02,                         //     INPUT (Data,Var,Abs)
    0x95, 0x01,                         //     REPORT_COUNT (1)
```

```
    0x81, 0x03,                         //     INPUT (Cnst,Ary,Abs)
    0xc0,                               //   END_COLLECTION
    0xc0,                               // END_COLLECTION
```

Supporting Usages in Multi-touch Digitizer Drivers (Windows 7)

Article • 01/18/2019

In the context of Windows Touch, *multi-touch* refers to support of a two or more trackable contact points. This topic outlines required and optional usages for a multi-touch digitizer driver. If your digitizer device supports only a single contact point, see Supporting Usages in Touch Digitizer Drivers.

Usage identifier values are defined in the Device Class Definition for HID 1.11 .

Required and Optional HID Usages

The report descriptor for a multi-touch digitizer must specify that the device is a HID touch screen (page 0x0D, usage 0x04).

In addition to the existing HID touch usages, multi-touch digitizer drivers must implement the following usages:

- X (page 0x01, usage 0x30) and Y (page 0x01, usage 0x31)
- Contact identifier (page 0x0D, usage 0x51)
- Tip switch (page 0x0D, usage 0x42)
- In-range (page 0x0D, usage 0x32)
- Contact count maximum (page 0x0D, usage 0x55)

The following usages are optional, but multi-touch digitizer drivers should implement them if the digitizer hardware supports them. These usages were ratified in the Windows Vista timeframe:

- Confidence (page 0x0D, usage 0x47)
- Width and height (page 0x0D, usages 0x48 and 0x49)
- Pressure (page 0x0D, usage 0x30)

HID Additions to Support Multi-touch

The HID Usage Tables define the following usages for multi-touch input from digitizers. Windows 7 supports these multi-touch usages, and vendors should implement them in

devices and drivers.

Expand table

Name	Description	Page	Type	ID
Contact identifier	Contact identifier	Digitizer	Dynamic Value (DV)	0x51
Configuration	Configuration	Digitizer	Collection Application (CA)	0x0E
Device mode	Input mode	Digitizer	DV	0x52
Device settings	Device settings	Digitizer	Collection Logical (CL)	0x23
Device Identifier	Device index	Digitizer	SV/DV	0x53
Contact count	Actual contact count	Digitizer	DV	0x54
Contact count maximum	Maximum number of contacts supported	Digitizer	DV	0x55

- Contact identifier

  Specifies the identifier of the current contact. An identifier must remain constant while the contact is detected by the device. Each separate concurrent contact must have a unique identifier. Identifiers can be reused if a contact is no longer detected. If the device supports "in-air" packets (the contact is hovering above the surface), the identifier must persist from the time that the contact is detected until the time that it goes out of range. In the report descriptor in the EloMT sample, the comment for this usage is "Temp Identifier."

- Configuration

  The Collection Application for the top-level collection that contains the feature report.

- Device Mode

  A read/write value feature to get and set the current input configuration of a device. In the EloMT sample, the comment for this usage is "Input Mode."

- Device settings

The logical collection that contains the device configuration usages (Device Identifier and Device Mode).

- Device Identifier

  The top-level collection for which the configuration is intended. Use Device Identifier if the report descriptor contains more than one multiple input top-level collection. For more information that is specific to this scenario, see Using Report Descriptors to Support Capability Discovery.

- Contact count

  Specifies the number of valid contacts in the current packet. Drivers that use parallel or hybrid mode should include this usage. A device that cannot provide this value must use **NULL** for all values in the first position that do not contain valid contact information.

- Contact count maximum

  Specifies the total number of contacts that a multi-touch device supports. This usage must be included in the multi-touch top-level collection and not in any child collection.

  The vendor-supplied driver may be queried dynamically for this value.

You can see examples of the previous usages in the EloMT sample in the WDK.

Selecting Packet Reporting Modes in Multi-touch Drivers (Windows 7)

Article • 01/18/2019

Windows supports three ways of reporting multi-touch data to the system: serial mode, parallel mode, and hybrid mode. The vendor-supplied HID report descriptor differs, depending on the mode that was selected.

We recommend that vendors use hybrid or parallel mode, because these modes can improve the efficiency of data delivery from operating system component to applications.

Serial Mode

Each packet contains information that describes a single physical contact point. Multiple contacts are streamed serially.

In this mode, devices report all contact information in a series of packets. Each packet contains information that describes a single physical contact. The device sends a separate packet for each concurrent contact.

For example, if two fingers are down, a device that uses serial mode sends an update for the first contact, and then sends an update for the second contact. This process repeats as long as both fingers are in contact with the digitizer.

Serial mode might reduce the effective reporting speed for each physical contact on the device. For example, if a device can send an update one time every millisecond (ms) and two physical contacts are present, each contact point updates only every 2 ms.

For a sample serial mode report descriptor, see Sample Report Descriptor (Serial Reporting Mode).

In comparison, hybrid and parallel reporting modes benefit from reduced data delivery overhead.

Parallel Mode

In parallel mode, devices report all contact information in a single packet. Each physical contact is represented by a logical collection that is embedded in the top-level collection. This logical collection contains all the usages that the device supports for each contact (for example, X, Y, and pressure). Because the device generally reports

fewer contacts than the maximum, the number of contacts that are reported in a parallel packet should be communicated either in the contact count usage or by setting null values for all invalid contacts in a packet.

Consider a device that supports three contacts. If the user currently has only two fingers on the digitizer, the parallel packet has only two valid contact data in a packet that can carry data for three contacts. In this case, the contact count should be set to two so that the client application knows that any information about more than two contacts is not valid.

Alternatively, the device can set the values of the contact usages beyond the second entry to null values. The client application can detect the actual contacts either by examining the value of the contact count usage or by reading the data until it encounters a null value.

For a sample parallel mode report descriptor, see Sample Report Descriptor (Parallel/Hybrid Mode).

A disadvantage of reporting multiple contacts in one packet is that space is wasted per packet every time that there are fewer contacts to report than the maximum number of contacts possible. Devices can use the hybrid reporting mode to reduce this inefficiency.

Hybrid Mode

In hybrid mode, the number of contacts that can be reported in one packet is less than the maximum number of contacts that the device supports. For example, a device that supports a maximum of 48 concurrent physical contacts can set up its top-level collection to report a maximum of 12 contacts in one packet. If 48 contact points are currently valid, the device can break these down into 4 serial packets that report 12 contacts each.

When a device reports data in this manner, the contact count usage value in the first packet should reflect the total number of contacts that are being reported in the hybrid packets. The other serial packets should have a contact count of 0. Using the previous example, the contact count usage in the first packet has a value of 48, whereas the latter three packets have a contact usage count of 0.

Hybrid and parallel reporting modes have the advantage of reducing the data delivery overhead.

For a sample hybrid mode report descriptor, see Sample Report Descriptor (Parallel/Hybrid Mode).

Null Values

Null values should be specified as outlined in the HID specification. The null bit must be set on all main items in the report descriptor. Be aware that a device can use either the contact count usage or null values to notify the host of the actual number of valid contacts in a packet.

Sample Report Descriptor - Serial Reporting Mode (Windows 7)

Article • 01/24/2019

The following example shows a sample report descriptor for serial reporting mode. It contains a top-level collection with a single embedded physical collection:

```
    0x05, 0x0d,                    // USAGE_PAGE (Digitizers)
    0x09, 0x04,                    // USAGE (Touch Screen)
    0xa1, 0x01,                    // COLLECTION (Application)
    0x85, REPORTID_MTOUCH,         //   REPORT_ID (Touch)
    0x09, 0x22,                    //   USAGE (Finger)
    0xa1, 0x00,                    //   COLLECTION (Physical)
    0x09, 0x42,                    //     USAGE (Tip Switch)
    0x15, 0x00,                    //     LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //     LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //     REPORT_SIZE (1)
    0x95, 0x01,                    //     REPORT_COUNT (1)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x95, 0x03,                    //     REPORT_COUNT (3)
    0x81, 0x03,                    //     INPUT (Cnst,Ary,Abs)
    0x09, 0x32,                    //     USAGE (In Range)
    0x09, 0x47,                    //     USAGE (Confidence)
    0x95, 0x02,                    //     REPORT_COUNT (2)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x95, 0x0a,                    //     REPORT_COUNT (10)
    0x81, 0x03,                    //     INPUT (Cnst,Ary,Abs)
    0x05, 0x01,                    //     USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x7f,              //     LOGICAL_MAXIMUM (32767)
    0x75, 0x10,                    //     REPORT_SIZE (16)
    0x95, 0x01,                    //     REPORT_COUNT (1)
    0x65, 0x00,                    //     UNIT (None)
    0x09, 0x30,                    //     USAGE (X)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x09, 0x31,                    //     USAGE (Y)
    0x46, 0x00, 0x00,              //     PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x05, 0x0d,                    //     USAGE PAGE (Digitizers)
    0x09, 0x48,                    //     USAGE (Width)
    0x09, 0x49,                    //     USAGE (Height)
    0x95, 0x01,                    //     REPORT_COUNT (2)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x81, 0x03,                    //     INPUT (Cnst,Ary,Abs)
    0x09, 0x51,                    //     USAGE (Contact Identifier)
    0x75, 0x10,                    //     REPORT_SIZE (16)
    0x95, 0x02,                    //     REPORT_COUNT (1)
    0x81, 0x02,                    //     INPUT (Data,Var,Abs)
    0x09, 0x55,                    //    USAGE(Maximum Count)
```

```
    0x15, 0x00,                    //   LOGICAL_MINIMUM (0)
    0x25, 0x08,                    //   LOGICAL_MAXIMUM (255)
    0x75, 0x08,                    //   REPORT_SIZE (8)
    0x95, 0x01,                    //   REPORT_COUNT (1)
    0xb1, 0x02,                    //   FEATURE (Data,Var,Abs)
    0xc0,                          //   END_COLLECTION
    0xc0,                          // END_COLLECTION
```

Sample Report Descriptor - Parallel/Hybrid Mode (Windows 7)

Article • 01/24/2019

This sample descriptor can serve as a starting point for creating a parallel or hybrid report descriptor. The mode used depends on the relationship between the maximum count and the contact count.

A hybrid report contains a contact count that is greater than the number of contacts that fit into one packet. A parallel report contains all contact information in a single packet. For more information, see Selecting Packet Reporting Modes in Multi-touch Drivers.

This report descriptor has a top-level collection with two embedded logical collections. Each logical collection represents data that can be received from a single physical contact. Be aware that the contact count usage is not contained in either logical collection; instead, it follows as part of the top-level collection.

```
    0x05, 0x0d,                    // USAGE_PAGE (Digitizers)
    0x09, 0x04,                    // USAGE (Touch Screen)
    0xa1, 0x01,                    // COLLECTION (Application)
    0x85, REPORTID_PEN,            //   REPORT_ID (Touch)
    0x09, 0x22,                    //   USAGE (Finger)
    0xa1, 0x02,                    //     COLLECTION (Logical)
    0x09, 0x42,                    //       USAGE (Tip Switch)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //       REPORT_SIZE (1)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x09, 0x32,                    //       USAGE (In Range)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x09, 0x47,                    //       USAGE (Confidence)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x05,                    //       REPORT_COUNT (5)
    0x81, 0x03,                    //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                    //       REPORT_SIZE (8)
    0x09, 0x51,                    //       USAGE (Contact Identifier)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x01,                    //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x7f,              //       LOGICAL_MAXIMUM (32767)
    0x75, 0x10,                    //       REPORT_SIZE (16)
    0x55, 0x00,                    //       UNIT_EXPONENT (0)
    0x65, 0x00,                    //       UNIT (None)
```

```
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0x00, 0x00,              //       PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x09, 0x31,                    //       USAGE (Y)
    0x46, 0x00, 0x00,              //       PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0xc0,                          //    END_COLLECTION
    0xa1, 0x02,                    //    COLLECTION (Logical)
    0x05, 0x0d,                    //     USAGE_PAGE (Digitizers)
    0x09, 0x42,                    //       USAGE (Tip Switch)
    0x15, 0x00,                    //       LOGICAL_MINIMUM (0)
    0x25, 0x01,                    //       LOGICAL_MAXIMUM (1)
    0x75, 0x01,                    //       REPORT_SIZE (1)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x09, 0x32,                    //       USAGE (In Range)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x09, 0x47,                    //       USAGE (Confidence)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x95, 0x05,                    //       REPORT_COUNT (5)
    0x81, 0x03,                    //       INPUT (Cnst,Ary,Abs)
    0x75, 0x08,                    //       REPORT_SIZE (8)
    0x09, 0x51,                    //       USAGE (Contact Identifier)
    0x95, 0x01,                    //       REPORT_COUNT (1)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x05, 0x01,                    //       USAGE_PAGE (Generic Desk..
    0x26, 0xff, 0x7f,              //       LOGICAL_MAXIMUM (32767)
    0x75, 0x10,                    //       REPORT_SIZE (16)
    0x55, 0x00,                    //       UNIT_EXPONENT (0)
    0x65, 0x00,                    //       UNIT (None)
    0x09, 0x30,                    //       USAGE (X)
    0x35, 0x00,                    //       PHYSICAL_MINIMUM (0)
    0x46, 0x00, 0x00,              //       PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0x09, 0x31,                    //       USAGE (Y)
    0x46, 0x00, 0x00,              //       PHYSICAL_MAXIMUM (0)
    0x81, 0x02,                    //       INPUT (Data,Var,Abs)
    0xc0,                          //    END_COLLECTION
    0x05, 0x0d,                    //    USAGE_PAGE (Digitizers)
    0x09, 0x54,                    //    USAGE (Contact Count)
    0x95, 0x01,                    //    REPORT_COUNT (1)
    0x75, 0x08,                    //    REPORT_SIZE (8)
    0x15, 0x00,                    //    LOGICAL_MINIMUM (0)
    0x25, 0x08,                    //    LOGICAL_MAXIMUM (8)
    0x81, 0x02,                    //    INPUT (Data,Var,Abs)
    0x09, 0x55,                    //    USAGE(Contact Count Maximum)
    0xb1, 0x02,                    //    FEATURE (Data,Var,Abs)
    0xc0,                          // END_COLLECTION
```

Implementing Top-Level Collections in Multi-touch Drivers (Windows 7)

Article • 01/18/2019

The report descriptor for a multiple input device must include at least one top-level collection for the primary device and a separate top-level collection for the mouse.

If your device supports versions of Windows earlier than Windows 7, the report descriptor must also include a top-level collection that contains a feature report that can be used to configure the device as a multiple input device, single-input device, or mouse device. For more information, see Using Report Descriptors to Support Capability Discovery.

For example, in the EloMT sample, the EloMT.c file includes a report descriptor that contains three top-level collections: the first contains two logical collections (one for each of the two supported physical contacts), the second is the feature report (also known as a configuration collection), and the third specifies usages for mouse support. Be aware that this driver can still support more than two concurrent contacts by using Hybrid mode. For information about how to select modes, see Selecting Packet Reporting Modes in Multi-touch Drivers.

On Windows 7, HID reports that supply information from multiple inputs must specify Collection Application (CA) 0x4 (Touch Screen) on the digitizer usage page.

Devices can send multi-touch data by using one report for each contact, or they can use the reporting modes that were outlined in Selecting Packet Reporting Modes in Multi-touch Drivers for a more efficient way to report data. Report descriptors for touch devices should use Finger (0x22) CL (Collection Logical) to group the data and control usages in the top-level collections, whereas the Stylus (0x20) CL should be used to group pen-related control and data usages.

Configuration collection enables you to configure devices to work with earlier versions of Windows. You can also use configuration collections to operate your touch device in different modes. For example, on Windows Vista, your device can default to single touch. On Windows XP, your device can assume mouse functionality. For information about how to access the configuration collection, see the subtopic "Feature Report Exclusivity" in Using Report Descriptors to Support Capability Discovery.

Using Report Descriptors to Support Capability Discovery (Windows 7)

Article • 01/24/2019

This section describes how vendors use the report descriptor to support capability discovery for touch and multi-touch devices.

Report Descriptor

A vendor-supplied driver reports its device capabilities to the operating system by providing a report descriptor. For a full example of a report descriptor, see the Elotouch.c file, which is part of the EloMT sample in the Windows Driver Kit (WDK).

For Windows 7 to detect a device's ability to support multiple inputs, the driver must include the contact identifier usage (0x51) in the report descriptor. Be aware that in Sample Report Descriptor (Serial Reporting Mode), this usage is in the single top-level physical collection, whereas in Sample Report Descriptor (Parallel/Hybrid Mode), this usage appears one time in both the logical collections that describe the multiple inputs.

Feature Report Exclusivity

In Windows 7, the system opens exclusively the configuration top-level collection that contains the device mode feature report. Because the operating system opens the feature report exclusively, the report is not accessible to third-party applications.

Because Windows 7 configures the device to report data by multiple input only, the top-level collection must support the required multi-touch usages. For information about required usages, see Supporting Usages in Multi-touch Digitizer Drivers.

In Windows XP and Windows Vista, third-party applications can use the feature report to select the currently active input mode, for example single touch or mouse-based input. We recommend single touch for Windows XP Tablet PC Edition and Windows Vista. We recommend the mouse for Windows XP and Microsoft Windows 2000.

Feature Report Requirements

The feature report must be in its own top-level collection and must include the multiple input configuration usages.

The following example shows a feature report from Elotouch.c:

```
    0x09, 0x0E,                    // USAGE (Device Configuration)
    0xa1, 0x01,                    // COLLECTION (Application)
    0x85, REPORTID_FEATURE,        //   REPORT_ID (Configuration)
    0x09, 0x23,                    //   USAGE (Device Settings)
    0xa1, 0x02,                    //   COLLECTION (logical)
    0x09, 0x52,                    //    USAGE (Device Mode)
    0x09, 0x53,                    //    USAGE (Device Identifier)
    0x15, 0x00,                    //    LOGICAL_MINIMUM (0)
    0x25, 0x0a,                    //    LOGICAL_MAXIMUM (10)
    0x75, 0x08,                    //    REPORT_SIZE (8)
    0x95, 0x02,                    //    REPORT_COUNT (2)
    0xb1, 0x02,                    //   FEATURE (Data,Var,Abs
    0xc0,                          //   END_COLLECTION
    0xc0,                          // END_COLLECTION
```

Device Mode

The Device Mode usage (0x52) can have one of the following values:

Mode	Value
Mouse (recommended default)	0x00
Single-input (single touch or stylus)	0x01
Multiple input	0x02

When single-input mode is set, vendors may do one of the following:

- Send information about the first contact only. This can be implemented in the firmware or the device driver.
- Drop all information about other contacts in the HID minidriver. This approach reduces the logic that is required in the firmware. However, it is available only to implementers who choose to write a device driver.

When mouse mode is set, the firmware or device driver should route the data from the first contact that was detected by using the mouse top-level collection.

Choose the appropriate default device mode for your device based on your device's capabilities and the operating system versions that it supports. To provide backward compatibility with earlier versions of Windows, we recommend mouse mode as the default. With the default set to mouse mode, the device can work with any operating system.

If you can guarantee that your device will not be used on any version of Windows earlier than Windows Vista, it is better to set default to single-input mode. Windows 7 reconfigures the device for multiple input if it discovers the capability.

Device Identifier

Device Identifier (0x53) is a static value (SV) when it is part of a digitizer or mouse top-level collection. It is required when a report descriptor contains multiple digitizer top-level collections of the same kind. This usage uniquely identifies the digitizer top-level collection and should appear in the feature report.

If the device can function as a mouse, the mouse collection should have the same device identifier as the corresponding digitizer collection. Devices with only one digitizer top-level collection are not required to specify a device identifier usage.

When the usage is part of a device settings logical collection, it is a dynamic value (DV). In this scenario, the usage enables the host to select the device that it wants to configure. A value of zero indicates all collections. A nonzero value indicates the top-level collection with matching device identifier.

Touch and Pen Support

If your device includes a digitizer that provides both Windows Touch and pen functionality, you must report touch and pen collections separately. If your driver might be installed on a slate Tablet PC or other computer that has no mouse, you should also report a mouse collection.

Windows Touch Samples (Windows 7)

Article • 08/24/2021

Windows Touch Samples

The legacy Windows touch samples are available in the Windows Driver Kit 7.1.0.

Sample name	Build environment	Target operating system	PnP driver	In-box driver	Sample description
Elotouch Driver	Windows 7 Windows Server 2008 Windows Vista Windows Server 2003	Windows 7 Windows Server 2008 Windows Vista Windows Server 2003 Windows XP	No	No	The Elotouch driver is a KMDF driver that can function as both a multi-touch driver and a mouse driver.
WacomKMDF Driver	Windows 7 Windows Server 2008 Windows Vista Windows Server 2003	Windows 7 Windows Server 2008 Windows Vista Windows Server 2003 Windows XP	No	No	The WacomKMDF driver is a KMDF version of the sample Wacom HID minidriver.

Elotouch Driver

Article • 06/21/2022

Description

The Elotouch driver is a KMDF driver that can work as both a multi-touch driver and a mouse driver. Its architecture is very similar to the WacomKMDF Driver.

The sample drivers share code that is located in the following directories:

- The src\input\hiddigi\common directory contains code that is common to the KMDF sample drivers.
- The src\input\hiddigii\util directory contains utility functions that are common to all the HID input sample drivers.

Building the Sample

Use the standard driver build tools in the Windows Driver Kit. In a driver directory, type **build**. The build script will generate the Elotouch.sys driver.

Installation

Driver Files

- EloMT.inf
- Elotouch.sys
- Hidkmdf.sys (build from the src\hid\hidusbfx2\hidmapper directory)
- WDF co-installer dll from <WDK ROOT>\redist\wdf\<platform>\

To install the drivers, copy the driver files (the .sys and .dll files) and the INF files to the same location. In Device Manager, complete the following steps.

On Windows XP (The driver works as a mouse on Windows XP and Windows Vista):

1. Right-click the device, and click **Update Driver**.
2. Select **Install from a list or specific location (Advanced)**, and then click **Next**..
3. Select **Don't search. I will choose the driver to install**, and then click **Have Disk**..

4. In the **Install From Disk** dialog box, type the path of the directory where you copied the driver and INF file, and then click **OK**..
5. Click **Finish**.

On Windows Vista and Windows 7

1. Right-click the device, and click **Update Driver**.
2. Click the **Browse my computer for driver software** link.
3. Click the **Let me pick from a list of device drivers on my computer** link.
4. Click **Have Disk**.
5. Navigate to the location of the driver file, and click the INF file.
6. Click **OK**.

Code Tour

This section includes a file manifest of all the files in the src\input\hiddigi directory.

File Manifest

src\input\hiddigi \ wacompen

File	Description
Errcodes.mc	Contains event code and messages.
Pch.h	Precompiled header file.
Sources	WDK sources file.
Makefile	WDK build environment make file.
Elotouch.c	Contains the OEM specific code.
Oempen.c	Contains the OEM specific code.
Elotouch.h	Contains the OEM specific definitions.
Elotouch.rc	The resource file for the driver.
EloMT.inx	INX file that is used to generate the INF file

src\input\hiddigi\common

File	Description
hid.c	Handles all the internal HIDClass IOCTLs.
naturalInput.h	Contains common definitions for UART digitizer drivers.
Pnp.c	Handles Plug and Play (PnP) and power management.
Serial.c	Contains all functions that deal with the serial port.
Serial.h	Contains serial port definitions.

src\input\hiddigi\util

File	Description
Errlog.c	Contains all the error logging functions.
Errlog.h	Contains error logging definitions.
Wtrace.h	Definitions for trace macros. This should be edited to enable tracing if needed.

WacomKMDF Driver

Article • 06/21/2022

Description

The WacomKMDF directory contains a KMDF version of the sample Wacom HID minidriver.

The WacomPen drivers are HID minidrivers for pen devices from Wacom Technology Corporation. These are electromagnetic input devices that use a 16550 UART-compatible interface. If such a device is installed on a computer system, the corresponding driver is loaded on top of the system-supplied Serial.sys driver, which acts as a lower-level device filter driver. The corresponding sample INF file can be generated from Wacompen.inx. This INF file installs the WacomPen driver for devices whose device ID is ACPI\WACF004. To install a device of this type by using Wacompen.inf, you must specify a device ID of ACPI\WACF004 in the ACPI BIOS.

The sample drivers share code that is located in the following directories:

- The src\input\hiddigi\common directory contains code that is common to the WacomPen and EloMT sample drivers.
- The src\input\hiddigi\util directory contains utility functions that are common to all the sample drivers. The utility functions handle tracing driver operation and error logging.

The sample driver demonstrates how to write a KMDF driver for a HID device even though KMDF does not natively support minidrivers. This is achieved by a WDM shim driver (Hidkmdf.sys) that acts as the HIDCLASS minidriver while the real driver is a lower filter in the driver stack. Otherwise, the sample is very similar to the WDM version in terms of how easy it is to customize for a new driver project.

Building the Sample

Use the standard driver build tools from the Windows Driver Kit. In a driver directory, type **build**. The build script will generate the Wacomdigi.sys driver.

Installation

Driver Files:

- Wacompen.inf
- Wacomdigi.sys
- Hidkmdf.sys (build from the src\hid\hidusbfx2\hidmapper directory)
- WDF co-installer from <WDK ROOT>\redist\wdf\<platform>\

To install the drivers, copy the driver files and the INF files to the same location. In Device Manager, complete the following steps:

On Windows XP Tablet PC Edition:

1. Right-click the device, and click **Update Driver**.
2. Select **Install from a list or specific location (Advanced)**, and then click **Next**..
3. Select **Don't search. I will choose the driver to install**, and then click **Have Disk**..
4. In the **Install From Disk** dialog box, type the path of the directory where you copied the driver and INF file, and then click **OK**..
5. Click **Finish**.

On Windows Vista and Windows 7:

1. Right-click the device, and then click **Update Driver**.
2. Click the **Browse my computer for driver software** link.
3. Click the **Let me pick from a list of device drivers on my computer** link.
4. Click **Have Disk**.
5. Navigate to the location of the driver file, and then click the INF file.
6. Click **OK**.

Code Tour

This section includes a file manifest of all the files in the src\input\hiddigi directory.

File Manifest

src\input\hiddigi\WacomKMDF

File	Description

File	Description
Errcodes.mc	Contains event code and messages.
Pch.h	Precompiled header file.
Sources	WDK sources file.
Makefile	WDK build environment make file.
WacomPen.c	Contains the OEM specific code.
Oempen.c	Contains the OEM specific code.
WacomPen.h	Contains the OEM specific definitions.
WacomPen.rc	The resource file for the driver.
Wacompen.inx	INX file that is used to generate INF files.

src\input\hiddigi\common

File	Description
hid.c	Handles all the internal HIDClass IOCTLs.
naturalInput.h	Contains common definitions for UART digitizer drivers.
Pnp.c	Handles PnP and power management.
Serial.c	Contains all functions that deal with the serial port.
Serial.h	Contains serial port definitions.

src\input\hiddigi\util

File	Description
Errlog.c	Contains all the error logging functions.
Errlog.h	Contains error logging definitions.
Wtrace.h	Definitions for trace macros. This should be edited to enable tracing if needed.

Additional Resources

Article • 11/05/2020

This topic presents resources for the legacy Windows Touchscreen and Pen devices for Windows 7 and earlier operating systems.

Device Certification and Distribution

Devices and, if applicable, drivers must be certified to gain access to the Windows Pointer Device features in the operating system.

After your multi-touch driver has qualified for the Windows logo, we recommend that you distribute your driver through Windows Update. For more information about how to distribute your driver, see Distributing Drivers on Windows Update.

For information about the Windows Logo Program, see the Windows Logo Program for Hardware Web site.

For guidance specific to Windows Touch in the Windows Logo Kit documentation, see Windows Touch Testing Overview. The Windows Logo Kit 1.4 contains tests and testing documentation for Windows Touch functionality.

Whitepapers

For information about how to write digitizer drivers for Windows 7, see Digitizer Drivers for Windows Touch and Pen-Based Computers.

For information about how to implement drivers for pen or touch digitizers in Windows Vista, see Pen and Touch Digitizer Drivers for Windows Vista.

For information about how to design multi-touch hardware, see How to Design and Test Multitouch Hardware Solutions for Windows.

HID Related Specifications

For the HID Device Class Definition and the HID usage tables, see the USB HID Information page.

For information about the HID over I2C protocol, please refer to the HID Over I2C Protocol Specification.

SDK Documentation

If you are writing a user-mode application that processes multi-touch input, see the Windows Touch SDK documentation.

Universal Serial Bus (USB)

Article • 06/21/2022

This document provides recommendations for the design and development of USB components. The goal of this document is to enable ecosystem partners to build a device with optimized USB functionality.

USB features in Windows 10

Windows 10 supports:

- Dual Role controllers, which can function as either a USB Host or a USB Device. For example, a phone can connect to a PC as a USB device or it can connect to other USB peripherals as a USB host.
- USB Type C, a flippable, reversible USB connector approximately the same size as a USB micro-B connector. In addition, USB Type-C provides support for the following features:
  - USB 3.1 Gen 2 (10Gb/s)
  - Power Delivery, enabling devices and systems to provide and consume up to 20V, 5A.
  - Alternate Modes, enabling non-USB protocols such as DisplayPort, Thunderbolt, or MHL to use the USB Type-C connector.
  - Billboard error message
- USB 2.0, 3.0, and now USB 3.1, allowing OEMs to easily choose from a wide variety of controllers and peripherals.
- Universal drivers can be built for USB peripherals that run on all Windows 10 Devices, from Internet-of-Things (IOT) to servers.

Partners are recommended to test these features by using the Hardware Lab Kit and to visit the USB Core Team blog for updated info about features and HLK tests, including these posts related to Windows 10:

- USB tests in the Windows 10 Hardware Lab Kit (HLK)
- New in Windows 10: USB Dual Role, Type-C, SuperSpeedPlus, and much more...
- New in Windows 10: USB Dual Role on Mobile

USB scenarios for Windows 10 devices

Windows 10 enables hardware vendors to innovate and build great Dual Role and USB Type-C systems. A few examples of the scenarios enabled by USB features in Windows 10 are illustrated below:

- Replace proprietary docking connectors with industry standard USB Type-C connector
- Faster charging via USB Type-C current and/or Power Delivery
- Allow for external USB peripherals to charge the system
- Output video/audio over USB Type-C using Alternate Modes
- Connect USB peripherals to Windows Mobile devices
- Write Universal Applications that can interact with custom USB peripherals

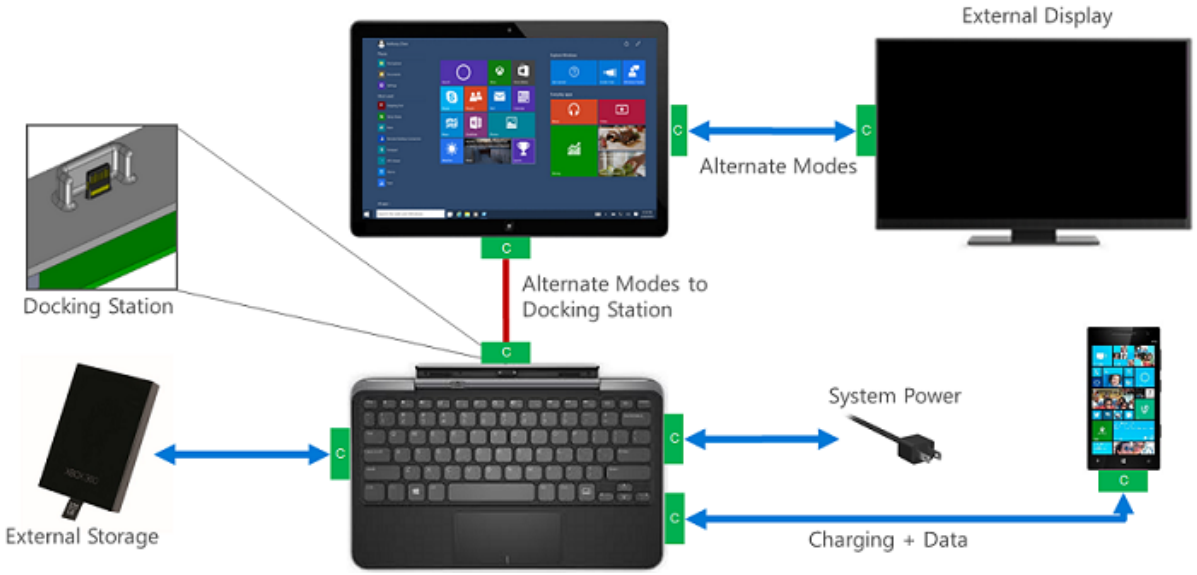


Figure 1 Example of new USB scenarios for a Windows 10 Desktop system

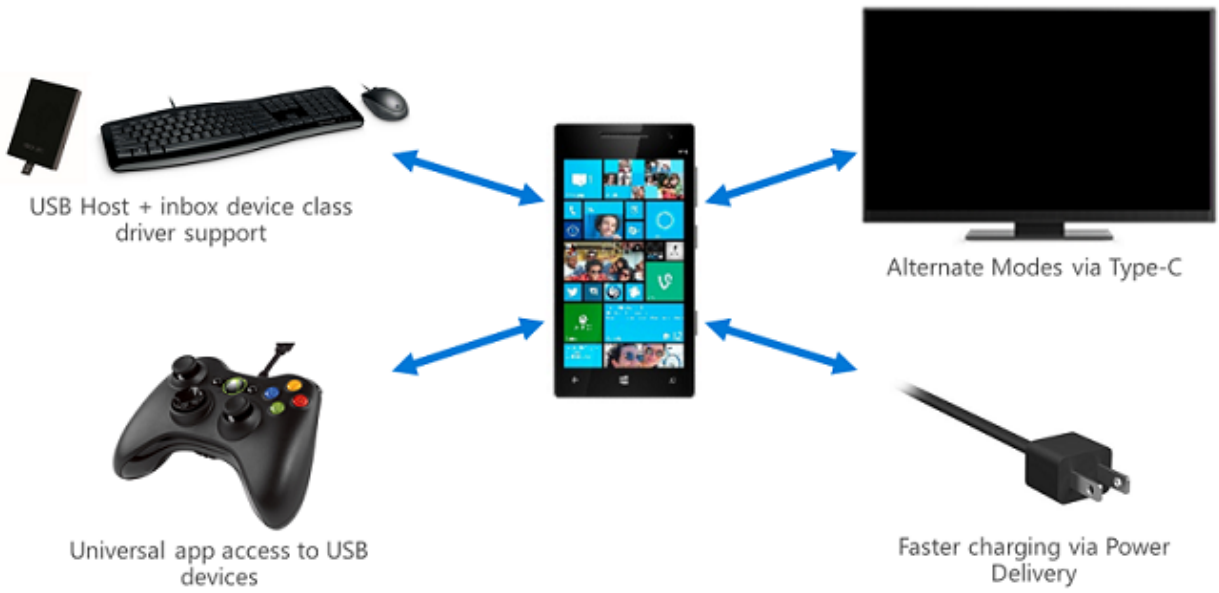


Figure 2 Example of new USB scenarios for a Windows 10 Mobile device

Guidelines for USB for Windows 10

- In general, USB should "just work" with minimal user interaction.
- Hardware/firmware is expected to make the initial policy decisions introduced with the Dual Role, Power Delivery, and Alternate Mode features.
- Windows will make policy adjustments to improve the user experience and help the user to troubleshoot issues if things go wrong.
- Partners should ensure their USB hardware (such as systems, hubs, cables, or accessories) can interoperate with other USB hardware coming to or already on the market. We encourage participation in USB-IF-hosted interoperability and compliance events.
- We recommend supporting the DisplayPort Alternate Mode for outputting video/audio over USB Type-C.
- For systems with multiple USB ports that have different USB capabilities, we recommend making it clear to the customer which ports have which capabilities. For example, an icon imprinted next to the USB port can be used to show that the port supports Alternate Modes and Power Delivery.
- Partners should follow the latest ACPI spec for how to properly describe their USB ports, such as the _UPC and _PLD methods.

Minimum hardware requirements for USB

USB is optional for all devices and computers that run Windows 10. Windows 10 supports the following USB controllers:

- Function controllers
- Host controllers
- Dual-role OTG controllers

For the complete set of hardware requirements for Windows 10, see Minimum hardware requirements.

Windows Hardware Compatibility Program specifications for USB

The Windows Hardware Compatibility Program leverages tests in the Windows Hardware Lab Kit (HLK), which OEMs can use to help diagnose issues early in the development process, ensure driver compatibility with Windows, and optionally certify

devices or systems. Tests in the HLK can validate Dual Role and Function mode on any Windows 10 edition.

The System.Fundamentals.SystemUSB.SystemExposesUSBPort Windows Hardware Compatibility Program requirement lists which USB controller types are recommended, supported, or not supported.

Maintained guidelines for USB from Windows 8.1

These guidelines from Windows 8.1 are maintained for Windows 10.

Hardware

For improved power efficiency and performance, it is recommended USB Host Controllers are at least USB 3.0 compatible with an XHCI controller integrated into the SoC or chipset. The operating system supports standard EHCI and XHCI 1.0 controllers, including debug registers. If the host controller is not fully compatible with the published standard specifications, the deviations must be documented and support for the host controller is determined on a case-by-case basis. In addition, the debug capability is important for XHCI host controllers.

USB Host Controller Interface	Remarks
XHCI 1.0+Errata or higher (including debug capability)	Required per the Windows HLK starting from June 2012
EHCI	Supported
UHCI/OHCI Companion Controllers	Not-supported

Related Resources

Blogs

USB Core Team blog

Windows Hardware Certification blog

Videos

Docking

Enabling New USB Connectivity Scenarios

Building New Apps for USB Accessories

Building great USB 3.0 devices

Integrating with the Windows Device Experience

Running Windows from an External USB drive with Windows To Go

Other Guides and Reference

Windows 10: What's new for USB

Universal Serial Bus (USB) Drivers

Setting Up a USB 3.0 Connection in Visual Studio

Setting Up a USB 3.0 Connection Manually

USB Reference

Universal Serial Bus 4 (USB4™)

Article • 09/06/2021

This section provides OEMs, IHVs and silicon vendors bringing up Windows systems supporting USB4 with information about the USB4 connection manager, facilitating debugging, and bring-up procedures.

Resources

- USB Core Team blog
- Windows Hardware Certification blog

See also

- Introduction to the USB4 connection manager in Windows
- USB4 design details and general requirements
- USB4 ACPI requirements
- USB4 power management requirements
- USB4 interdomain connections
- USB4 required testing
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

Introduction to the USB4 connection manager in Windows

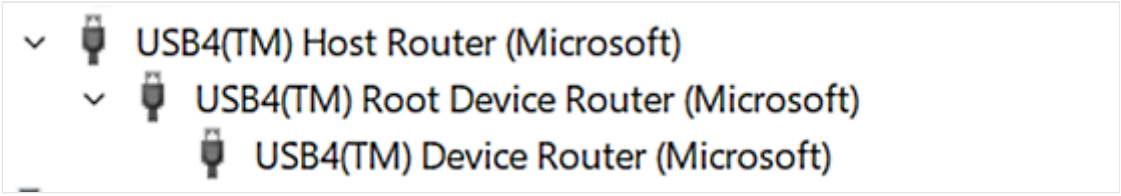
Article • 09/03/2021

The connection manager in a USB4 domain is responsible for enumeration, configuration, and management of the USB4 domain. It is responsible for:

- Initialization of the USB4 domain, including handoff from the UEFI/BIOS
- USB4 and Thunderbolt™ 3 router detection and configuration
- Path setup and teardown
- Bandwidth management among the configured tunnels
- Port management including lane bonding and disabling and TMU mode
- Support for host-to-host interface (P2P)
- Power state management

A single USB4 domain consists of one USB4 host router and its connected USB4 device routers. The USB4 host router device enumerates a virtual router known as the root device router, which is meant to abstract some components of the host router, just like a USB 3.x root hub does on a USB 3.x host controller.

In Windows Device Manager, when the view is changed to **Devices by connection** it appears as follows:



The **USB4 Host Router** device is associated with the service *Usb4HostRouter.sys*. The **USB4 Root Device Router** and any other **USB4 Device Router** is associated with the service *Usb4DeviceRouter.sys*.

Driver	Associated Device
USB4HostRouter.sys	USB4 host router
USB4DeviceRouter.sys	USB4 and Thunderbolt™ 3 hubs and devices
USB4P2PNetAdapter.sys	Virtual network adapter for supporting host to host connections

Starting with Windows 11, the use of the OS-provided connection manager drivers is required.

See also

- Universal Serial Bus 4
- USB4 design details and general requirements
- USB4 ACPI requirements
- USB4 power management requirements
- USB4 interdomain connections
- USB4 required testing
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

Universal Serial Bus 4 (USB4™) design details and general requirements

Article • 06/15/2023

In addition to the specification defined requirements, the following are some of the high-level design and user experience requirements.

- Devices that are tunneled over USB4 (USB 3.x, PCIe, and display), should work just as they would natively. No software changes should be required to the protocol stacks, and it should be transparent to them that they are tunneled over USB4.
- Partners should ensure their USB hardware (such as systems, hubs, cables, and accessories) can inter-operate with other USB hardware coming to or already on the market. We encourage participation in interoperability and compliance events hosted by the Universal Serial Bus-Implementers Forum (USB-IF).
- For systems with multiple USB-C® ports, Microsoft strongly recommends that they all support USB4 to provide a consistent user experience.
- Thunderbolt™ 3 compatibility is required per chapter 13 of the USB4 base specification and USB Type-C® Connector and Cable Specification. The Windows USB4 connection manager driver is designed to support Thunderbolt™ 3 peripherals attached to the USB4 host.
- For the first release of the USB4 connection manager on Windows 11, add-in cards are not supported.
- See complete set of detailed requirements from the USB4 Hardware Lab Kit (HLK) for the Windows Hardware Compatibility Program.
- The first release of the USB4 connection manager requires the host and device routers to support the USB4 specification and is implemented in accordance with USB4 Connection Manager Guide Rev 1.0rc. In addition to other ECNs, hardware implementers must ensure that they've reviewed the following ECNs as they're of high significance to the USB4 connection manager. This is not meant to be a complete list of ECNs required. These documents can be found in the USB4 specification download.
  - USB4 1.0 ECN - Change in ISC and ISS Registers Address.pdf
  - USB4 1.0 ECN - DROM Base Address.pdf
  - USB4 1.0 ECN - Router Ready Bit.pdf

- USB4 1.0 ECN - HI memory order.pdf
- USB4 1.0 ECN - Host Router State Machine.pdf
- USB4 1.0 ECN - Change C3S Bit Polarity.pdf
- USB4 1.0 ECN - Buffer Allocation Request Operation.pdf
- USB4 1.0 ECN - DP BW Allocation.pdf*

  *DP bandwidth allocation ECN is not required in its entirety. Only the connection manager ID, adapter number, and TopologyID features of the ECN are currently used by the Windows USB4 connection manager in its graphics power management scheme. For details, see Graphics driver requirements.

> **Note**
>
> Some implementation details of the connection manager are provided to aid in debugging. You should not take dependencies on implementation details.

USB Type-C® requirements

The Windows USB4 connection manager does not have any software dependencies on the USB Type-C® software stack. For example, UCMCx, UCSICx, and so on. The host router and device routers must follow the requirements described in the latest USB Type-C® connector specification for USB4 discovery and entry.

USB4 devices may provide a USB interface via a USB billboard device class when it cannot connect as a USB4 device. Windows presents users with a notification when the billboard device is discovered indicating that the device functionality maybe limited.

BIOS to operating system handoff

In the absence of a hardware interface for USB4 host router reset, Windows USB4 connection manager implements a host router configuration space cleanup that emulates a hardware-initiated host router reset. The objective of host router config space cleanup is to bring the configuration space values back to default in case there is a pre-boot connection manager (UEFI CM) running before Windows USB4 connection manager runs. This is possible if a UEFI connection manager runs before cold boot or resume from hibernate (S4).

Bandwidth sharing policy

The Windows USB4 connection manager provides bandwidth management capabilities to prevent oversubscription of a USB4 link. A maximum of 90 percent of the USB4 link bandwidth may be explicitly allocated for USB 3.x (isochronous), PCIe and DisplayPort™ traffic, divided according to the bandwidth sharing policy implemented by the connection manager.

When a device router is enumerated, the capacity of the USB4 link in both upstream and downstream directions is determined after lane bonding, resulting in a calculated link rate of 10Gbps, 20Gbps, or 40Gbps, for example. The available capacity of the link is then calculated as 90 percent of this. In the example, 90 percent would be 9Gbps, 18Gbps, or 36Gbps, respectively.

> **Note**
>
> For Thunderbolt™ 3 links, the following descriptions of USB 3.x & PCIe calculations do not apply. The full bandwidth is available for DisplayPort™ tunnels.

The link's bandwidth is initially assigned to both USB 3.x & PCIe. This combined USB 3.x & PCIe bandwidth is divided 2/3 to USB 3.x, and 1/3 to PCIe. The USB 3.x portion of this bandwidth is programmed to the USB 3.x host controller for isochronous usage, through a handshake process where the CMR bit in the USB 3.x adapter config space is set, and the USB4 connection manager awaits an acknowledgement from the USB 3.x host controller before programming the available bandwidth. Maximum USB 3.x bandwidth will likely be constrained to the USB 3.x link rate (10Gbps/20Gbps), accounted for when calculating bandwidth to assign.

DisplayPort™ bandwidth (DP BW) allocation mode

Starting in Windows 11 build 22621.1344 (KB5022913), and on systems supporting DisplayPort™ bandwidth (DP BW) allocation mode as defined in the USB4 specification, the USB4 connection manager allows for dynamic DisplayPort™ (DP) tunnel bandwidth allocation. This feature allows more efficient and dynamic use of the bandwidth available to DP tunnels through coordination with the graphics stack on the system to optimally divide bandwidth between all tunnels.

Upon creation of a DP tunnel, the USB4 connection manager will initially attempt to reserve bandwidth for the tunnel based on indicated capabilities of the DP IN and DP OUT adapters forming the tunnel. If sufficient bandwidth is not available, the initial reservation will be reduced as with non-DP BW allocation mode tunnels.

The connection manager will then check for DP BW allocation mode support for the DP adapter when the tunnel is created. If DP BW allocation mode is supported, it will be enabled for the DP IN adapter, the total available (including previously reserved) bandwidth will be written out to the "ESTIMATED_BW" register of the adapter, and allocated bandwidth will be written out to the "ALLOCATED_BW" register of the adapter as 0.

Once the DP tunnel has been created, any changes in available bandwidth for the tunnel (due to other DP tunnels or other bandwidth operations) will result in the USB4 connection manager updating the "ESTIMATED_BW" register of the DP IN adapter to notify it of the additional or reduced bandwidth available to it.

After the tunnel has been created, the graphics driver will send a notification to request the precise bandwidth necessary to support the selected resolution, refresh rate and bit-depth etc., of the tunneled monitor from the USB4 connection manager.

Thereafter, the graphics driver may request additional or reduced bandwidth due to changes in monitor performance requirements. If additional bandwidth is requested from the USB4 connection manager, the connection manager may attempt to reclaim bandwidth allocated to USB3 on the links shared with the tunnel. The USB4 connection manager will then update the DP IN adapter configuration space with the results of the operation, and any changes in the bandwidth available and/or assigned to it.

Systems without support for DisplayPort™ bandwidth allocation mode

On systems without support for the DisplayPort™ bandwidth allocation mode feature defined in the USB4 specification, DisplayPort™ (DP) bandwidth is then assigned on a first-come, first-served allocation basis. When a DP tunnel is created, the USB4 connection manager will examine in-use bandwidth for other DP tunnels, and query the USB 3.x host controller to determine how much bandwidth is currently in use for isochronous traffic, using the described handshake process. If bandwidth is available, it will be reclaimed in a sufficient amount for the DP tunnel, which is determined based on the capabilities of the DP IN and DP OUT adapters forming the tunnel.

If there is not sufficient bandwidth to satisfy the capabilities of the DP IN/OUT adapter pair, DP capabilities will be masked off to match the bandwidth available, potentially resulting in a lower than native maximum resolution for the DP tunnel. If enough bandwidth is not available to satisfy the lowest DP configuration, the DP tunnel will be failed. If there is insufficient bandwidth for the highest common configuration, and there are any previously created DP tunnels for which bandwidth has not been released

following polling, the creation of a new tunnel will be halted until bandwidth is potentially released by the previous tunnel.

Once a tunnel is successfully created, the connection manager will poll the DP IN adapter configuration space until common capabilities (reflecting actual bandwidth required for the output device) are available. At this point, if excess bandwidth is allocated, it will be released back to the combined USB 3.x & PCIe bandwidth pool, and be reprogrammed to the USB 3.x host controller.

Display considerations

Using tunneled displays in WinPE

Tunneled display requires a functional graphics driver running in the OS. If the OEM hasn't made graphics drivers available in WinPE (Windows Preinstallation Environment), such images will not be able to operate tunneled displays. When booting WinPE it is recommended to use the built-in display panel. If that is unavailable, then connect the external display in a non-tunneled configuration (native DP, HDMI, or USB Type-C Alt-Mode). If neither of these options are available, then the appropriate graphics driver package must be added to the WinPE image using DISM or equivalent mechanism.

Security considerations

Disabling PCIe tunneling

As a security feature, you may offer the ability to disable PCIe tunneling using the _OSC for USB. The Windows USB4 connection manager will not create PCIe tunnels when PCIe tunneling is disabled. Further, the Thunderbolt™ 3 device routers may fail to enumerate because in the absence of PCIe tunneling in Thunderbolt™ 3, USB functionality offered from the PCIe enumerated USB host controller is also impacted.

Consider disabling Thunderbolt™ 3 alternate mode when PCIe tunneling is disabled. This will allow a Thunderbolt™ 3 router or dock to fall back to USB 3.x operation and (DP) alt-mode operation.

DMA remapping policy for the USB4 host router

To limit the impact from invalid DMA accesses, DMA remapping is enabled for the USB4 host router device. The `DmaRemappingCompatible` setting is set to 1 so that DMA remapping policy is always enabled for the host router.

USB4 HLK requirements

The following is a complete list of USB4 requirements.

Device.BusController USB4 requirements

- USB4 Domain Sleep
- USB4 Host Router PCIe Support
- USB4 Host Router TBT3 Support
- USB4 Routers Using Microsoft USB4 Stack
- USB4 Host Router FPB Support
- USB4 USB Implementers Forum (IF) certification

System.Fundamentals USB4 requirements

- USB4 Systems BIOS Handoff Support
- USB4 Systems DP Alternate Mode Support
- USB4 Systems PCIe Tunneling Support
- USB4 Systems Thunderbolt 3 Compatibility
- USB4 Systems Microsoft USB Stack Support
- USB4 Systems Support All Type-C® Connectors
- USB4 Host Router Systems in ACPI Describe Unique IDs
- USB4 Systems Support OS Capabilities for USB
- USB4 Systems Graphics Driver Support
- USB4 Systems USB3 over USB4 Tunneled Protocol Mapping Support
- USB4 Systems PCIe Over USB4 Tunneled Protocol Mapping Support
- USB4 USB3 xHCI functionality over USB4

See also

- Universal Serial Bus 4
- Introduction to the USB4 connection manager in Windows
- USB4 ACPI requirements
- USB4 power management requirements
- USB4 interdomain connections
- USB4 required testing
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

USB4 ACPI requirements

Article • 04/27/2022

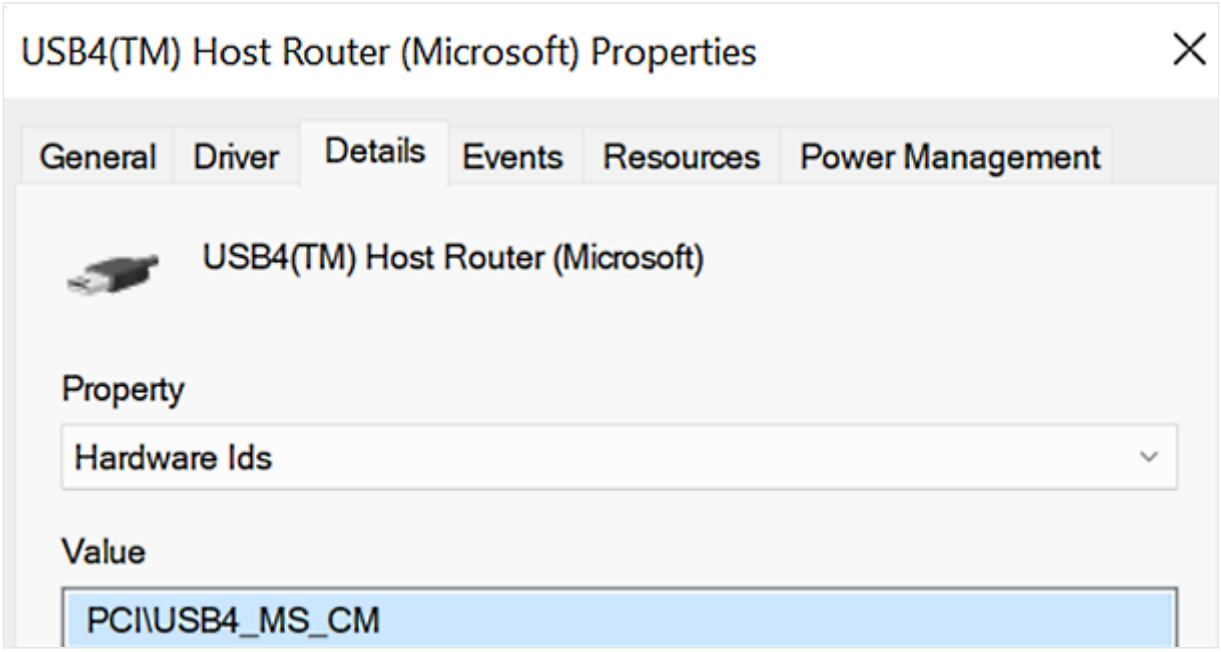
This article describes how USB4 devices interface with an Advanced Configuration and Power Interface (ACPI) device. ACPI devices are defined by the Advanced Configuration and Power Interface (ACPI) Specification.

_OSC (Operating System Capabilities) for USB4

The BIOS must grant control to the USB4 connection manager as per ACPI 6.4 specification. The system must grant control of native USB4 support in platform-wide operating system power management (OSPM) capabilities. Control is granted when _OSC is called by the operating system with *Query Flag* set to 0 and *Native USB4 Support* set to 1.

Additionally, _OSC for USB must also be implemented. The BIOS may disallow control over PCIe tunneling for security reasons as per required policies or user-settings. However, USB tunneling, DisplayPort™ tunneling and interdomain USB4 connections must always be enabled. The connection manager will place the device into a failed state if USB tunneling, DisplayPort™ tunneling or interdomain connections are disabled.

Once both capabilities are implemented as described above, the PCIe bus will add a hardware-ID of "PCI\USB4_MS_CM" to the list of hardware IDs for PCIe enumerated host routers. This will allow Plug and Play to load the USB4 connection manager driver.



Note

_OSC settings for native USB4 support or _OSC for USB should not be changed across a hibernate (S4) transition. A full reboot is required when such settings are altered.

Port mapping (_DSD) for USB 3.x and PCIe

For the PCIe and USB 3.x software stacks to establish power relations with the USB4 host router, device-specific data (_DSD) for the tunneled PCIe and USB 3.x ports is required. In the absence of this, the USB4 domain may power down without coordinating with the tunneled PCIe and USB 3.x devices.

> ⓘ **Note**
>
> The host interface instance must always end with a 3-bit digit (0-7).

The following is a port mapping example in ACPI source language (ASL):

ASL

```
Scope (\_SB.PCI0)
{
  Device (NHI0) { } //Host interface instance must always end with a 3-bit
digit (0-7).
  Device (DSB0) //Tunneled PCIe port instance
  {
    Name (_DSD, Package () {
      ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"), //Device Properties
UUID
      Package () {
        Package () { "usb4-host-interface", \_SB.PCI0.NHI0 },
        Package () { "usb4-port-number", PortInstance#}, // PortInstance# is
unused and can be 0
      }
    })
  }
…
  Device (SS01) // e.g. Tunnel capable USB 3.x super speed port under RHUB
  {
    …
    Name (_DSD, Package () {
      ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"), //Device Properties
UUID
      Package () {
        Package () { "usb4-host-interface", \_SB.PCI0.NHI0 },
        Package () { "usb4-port-number", PortInstance#}, // PortInstance# is
unused and can be 0
      }
```

```
    })
  }
```

Existing USB requirements for describing USB port capabilities(_UPC) remain unchanged for USB4.

Additional requirements for PCIe ports

PCIe root ports associated with USB4 must be designated as capable of handling hot plug events while in D3 state. For more information, see Identifying PCIe Root Ports supporting hot plug in D3 and Identifying externally exposed PCIe Root Ports.

PCIe root ports should be appropriately marked as capable of waking from D3Cold by using the `_S0W` and `_CRS`, or `_PRW` methods. For more information about `_S0W` and `_CRS`, or `_PRW`, see the ACPI specification and firmware requirements for D3cold.

Device-specific method to prevent data role swap for host-to-host connections

On certain platforms, Windows system UCSI OPM drivers may try to correct the USB data role in a USB connection by requesting a USB data role swap using the UCSI "Set USB Operation Role" (`SET_UOR`) command. However, this may impact the host-to-host connection, as either side of such a connection may initiate this role correction. To address this problem, the platform must implement ACPI _DSM method under the UCSI device as follows to claim that data role correction in UCSI OPM should be disabled.

USB device-specific method (_DSM)

To disable USB data role correction in UCSI OPM, Windows defines a device-specific method (_DSM) that has the function described below.

Function 4: Disable the USB data role correction in UCSI OPM

Arguments

Arg0: UUID = 6F8398C2-7CA4-11E4-AD36-631042B5008F

Arg1: Revision ID = 0

Arg2: Function index = 4

Arg3: Empty package (not used)

Return value

An integer containing one of the following values:

Element	Object type	Description
Disabled	4-byte (32-bit) unsigned long	Specify whether the role correction in UCSI OPM should be disabled. 0x00 (Default) - Do not disable the USB data role correction. 0x01 - Disable the USB data role correction in UCSI OPM.

When this function is used, the _DSM method must appear under the UCSI device.

> **Note**
>
> Function index 0 of every _DSM is a query function that returns the set of supported function indexes. It's always required. For more information, see section **9.1.1 _DSM (Device Specific Method)**, in the ACPI 6.4 specification.

See also

- Universal Serial Bus 4
- Introduction to the USB4 connection manager in Windows
- USB4 design details and general requirements
- USB4 power management requirements
- USB4 interdomain connections
- USB4 required testing
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

USB4™ power management requirements

Article • 10/04/2024

USB4 host routers must reliably support USB4 sleep and entry and exit. The tunneled protocols must not electrically signal disconnect to their respective tunnels on entry or exit from USB4 sleep. Additionally, the USB4 host router must replay the protocol wake over the appropriate tunnel after the USB4 domain has resumed from sleep.

The tunneled device protocols (PCIe, USB 3.x, and display) establish power relations with the associated USB4 host router device which allows the connection manager to initiate sleep entry when the tunneled devices are idle. The PCIe and USB 3.x device drivers starting in Windows 11 have the support to identify and power manage the associated USB4 host router device based on the ACPI based (_DSD) port mapping described in the USB4 ACPI requirements.

The graphics stack in Windows 11 has also been updated to identify the USB4 host router device and power manage the USB4 host router device. This requires support from the IHV provided graphics driver as described in Graphics driver requirements. The USB4 connection manager obtains the host router device ID from ACPI and writes to ADP_DP_CS_2.CM_ID in the DP-IN adapter configuration space. As described in the ACPI requirements, the host interface instance must always end with a 3-bit digit (0-7).

Graphics driver requirements

To ensure the correct power management associations, the graphics stack must be able to identify which USB4 DP-IN adapter to which the monitor is connected. This is achieved by the USB4 domain setting the DPCD register via the USB4 DP-IN adapter that the graphics stack can read. These requirements are defined in DisplayPort™ 2.0 specification (Errata).

Expand table

DPCD field name	DPCD field address	Note
USB4_Driver_ID	E000Fh [3:0]	Uniquely identifies the host router in the system. This is set by the USB4 connection manager writing obtaining the host router device ID obtained from

DPCD field name	DPCD field address	Note
		ACPI and writing it to ADP_DP_CS_2.CM_ID in the DP IN Adapter configuration space.
USB4_ROUTER_TOPOLOGY_ID	E001Bh - E001Fh	Uniquely identifies the USB4 device in the host router domain
DP_IN_Adapter_Number	E000Eh [5:0]	Uniquely identifies the DP-IN adapter on the USB4 device
DP Tunneling Support	E000Dh [0]	A value of 1 indicates USB4 tunneling

The graphics driver requirements are:

1. WDDM 3.0 or above driver
2. Driver exposes a new adapter cap to indicate USB4 support
3. All static video present network (VidPN) targets exposed by the driver need to be reported as power components
4. Each USB4 dynamic target connected to same static target must be connected to the same host router. That is, the USB4_Driver_ID must have the same value. Connecting a USB4 DisplayPort monitor to a display only or indirect display WDDM driver is not supported.

USB4 driver capabilities

The `SupportUsb4Targets` capability has been added to the DXGK_DISPLAY_DRIVERCAPS_EXTENSION structure that is queried from the driver at start time. Setting this value is only valid for WDDM 3.0 drivers and should only be set if the driver does meet all the requirements listed above.

Reporting USB4 monitor DPCD values

In WDDM 3.0 there is a new `DXGK_CONNECTION_CHANGE.MonitorConnect.MonitorConnectFlags` field that has a `Usb4DisplayPortMonitor` flag the driver uses to indicate this monitor is a USB4 connected monitor.

```
C++

typedef struct _DXGK_CONNECTION_MONITOR_CONNECT_FLAGS
{
```

```
    union
    {
        struct
        {
            UINT    Usb4DisplayPortMonitor  : 1;
            UINT    Reserved                :31;
        };
        UINT Value;
    };
} DXGK_CONNECTION_MONITOR_CONNECT_FLAGS;

typedef struct _DXGK_CONNECTION_CHANGE {
    ULONGLONG                       ConnectionChangeId;
    D3DDDI_VIDEO_PRESENT_TARGET_ID  TargetId            :24;
    DXGK_CONNECTION_STATUS          ConnectionStatus    : 4;
    UINT                            Reserved            : 4;
    union {
        struct {
            D3DKMDT_VIDEO_OUTPUT_TECHNOLOGY LinkTargetType;
            DXGK_CONNECTION_MONITOR_CONNECT_FLAGS MonitorConnectFlags;
        } MonitorConnect;
        struct {
            D3DKMDT_VIDEO_OUTPUT_TECHNOLOGY BaseTargetType;
            D3DDDI_VIDEO_PRESENT_TARGET_ID  NewTargetId;
        } TargetConnect;
        struct {
            D3DKMDT_VIDEO_OUTPUT_TECHNOLOGY BaseTargetType;
            D3DDDI_VIDEO_PRESENT_TARGET_ID  NewTargetId;
        } TargetJoin;
    };
} DXGK_CONNECTION_CHANGE, *PDXGK_CONNECTION_CHANGE;

typedef struct _DXGK_CONNECTION_USB4_INFO
{
    UINT Dpcd_DP_IN_Adapter_Number;
    UINT Dpcd_USB4_Driver_ID;
    BYTE Dpcd_USB4_ROUTER_TOPOLOGY_ID[5];
} DXGK_CONNECTION_USB4_INFO, *PDXGK_CONNECTION_USB4_INFO;

typedef struct _DXGKARG_QUERYCONNECTIONCHANGE
{
    DXGK_CONNECTION_CHANGE     ConnectionChange; // out: Buffer into which
the oldest available change is copied by driver
    PDXGK_CONNECTION_USB4_INFO pUsb4MonitorInfo; // inout: Pointer to an
operating system allocated structure driver should
                                                 //        complete if
MonitorConnect.Flags.Usb4DisplayPortMonitor was set
} DXGKARG_QUERYCONNECTIONCHANGE;
```

When a DisplayPort™ monitor has been connected to the specified video present network (VidPN) target and is connected to a USB4 DP-IN adapter the driver should:

1. Set `DXGK_CONNECTION_CHANGE.MonitorConnect.Flags.Usb4DisplayPortMonitor`

2. Complete the fields in `DXGKARG_QUERYCONNECTIONCHANGE.pUsb4MonitorInfo`

All static VidPN targets being reported as PoFx components

To enable correct power management between the graphics and USB stack, all static VidPN targets are required to be reported as power management framework (PoFx) components through `DxgkDdiQueryAdapterInfo(DXGKQAITYPE_NUMPOWERCOMPONENTS)` and `DxgkDdiQueryAdapterInfo(DXGKQAITYPE_POWERCOMPONENTINFO)`.

All dynamic targets created from same static VidPN target

USB4 monitors connected to dynamic VidPN targets are supported. But all USB4 dynamic VidPN targets connected to a static VidPN target must have the same `USB4_Driver_ID values`, they must be connected to the same USB4 host router.

CLx low power states

To save power and reduce thermal load, the connection manager enables first depth links to enter CL0s and CL1, if supported by hardware and currently tunneled protocols. Both the upstream facing port (UFP) and downstream facing port (DFP) must support CL0s and CL1 for the low power states to be enabled on the link. Subsequently, if the TMU accuracy requirements and tunneled activity on the link allow, the link may enter CL0s or CL1 to save power.

XHCI USB3 tunneling support

To save system power by keeping USB4 host routers idling longer, the XHCI controller associated with the USB4 host router must support the USB3 Tunneling Support ECN (Tunneled Mode support bit and USB3 Tunneling Support capability). An equivalent proprietary mechanism must be available if USB3 Tunneling Support ECN is not supported. The USB3 Tunneling Support ECN enables USB4 host routers to stay idle when a USB4-mapped USB3 port is operating in native mode.

See also

- Universal Serial Bus 4
- Introduction to the USB4 connection manager in Windows

- USB4 design details and general requirements
- USB4 ACPI requirements
- USB4 interdomain connections
- USB4 required testing
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

USB4 interdomain connections

Article • 09/03/2021

The Windows USB4 connection manager supports the Ethernet over USB4 interdomain protocol, also known as USB4NET. This enables two USB4 PCs to establish a network connection between each other when connected using a USB4 cable, akin to connecting an ethernet cable between network cards on two PCs. Since the protocol is backwards compatible with the protocol that was in use for Thunderbolt™ 3, this also works if one of the PCs is a Thunderbolt™ 3 PC, connected with a USB4 or Thunderbolt™ 3 cable.

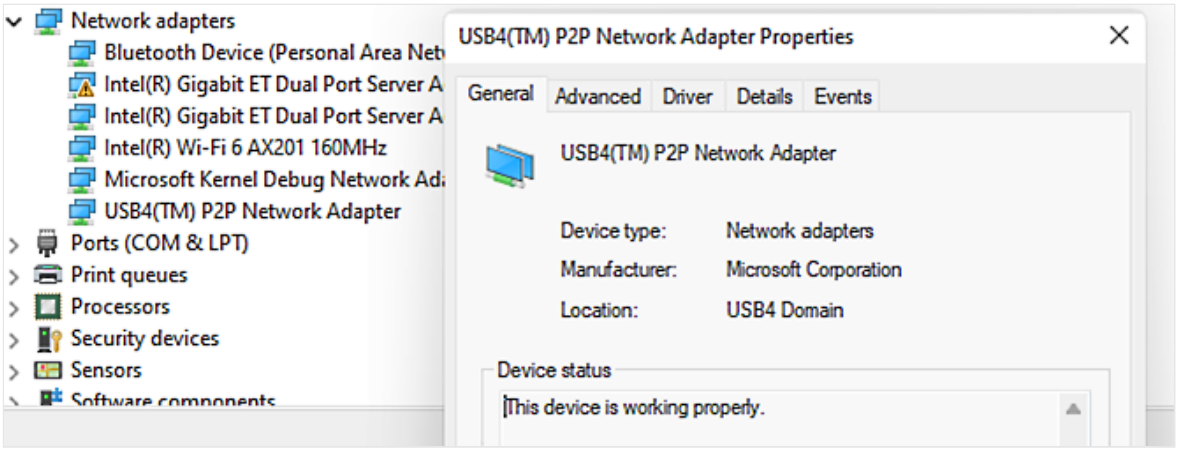
Windows ignores any loopback connections created by connecting two ports on the same system (or ports on hubs connected to the same system) with a USB4 cable. Only connections between two PCs are considered functional interdomain connections. Connections between two USB4 host routers on the same system, while technically interdomain since they are distinct USB4 domains, are also ignored by Windows since they do not enable any functional user experiences.

The software connection manager starting with Windows 11 enables file sharing over a USB4 interdomain connection using existing network file sharing techniques and user interfaces. When a connection is established, a new network adapter is enumerated and a link-local IPv4 address (169.254.0.0/16) is automatically assigned. The user may then share any folders using the File Sharing Wizard and access any shared folders by browsing to the IP address of the peer from within Windows Explorer. Using the computer name may not be guaranteed to use the USB4 connection if the remote computer is also visible over another network, like WiFi.

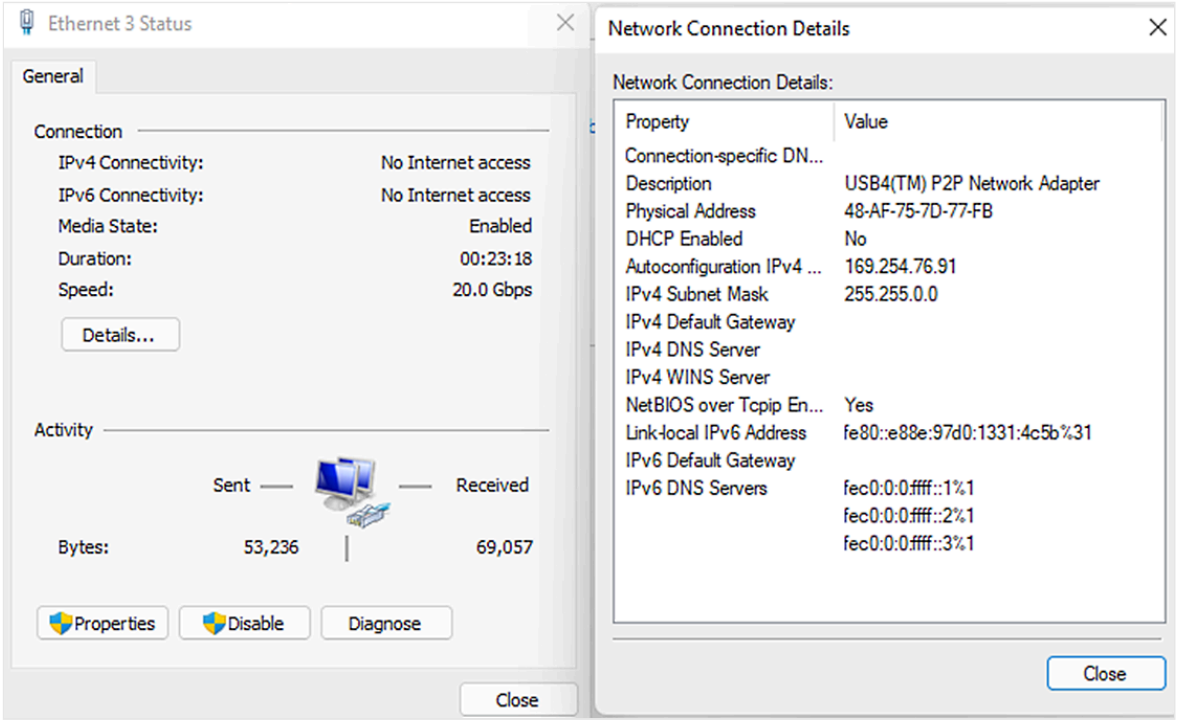
Steps to perform a file transfer over a USB4 interdomain connection

1. Connect the two PCs together with a USB4 cable.

2. A new network adapter appears on the system.



3. Open the network adapter properties and note the IP address of each system.



4. Ensure that firewall and network sharing settings allow for file sharing.

   Turn on **Public folder sharing** on the PC you are trying to access. This will allow you to access the *Users* folder of the user whose credentials you use to remotely login when presented the login prompt in Windows Explorer (see below).

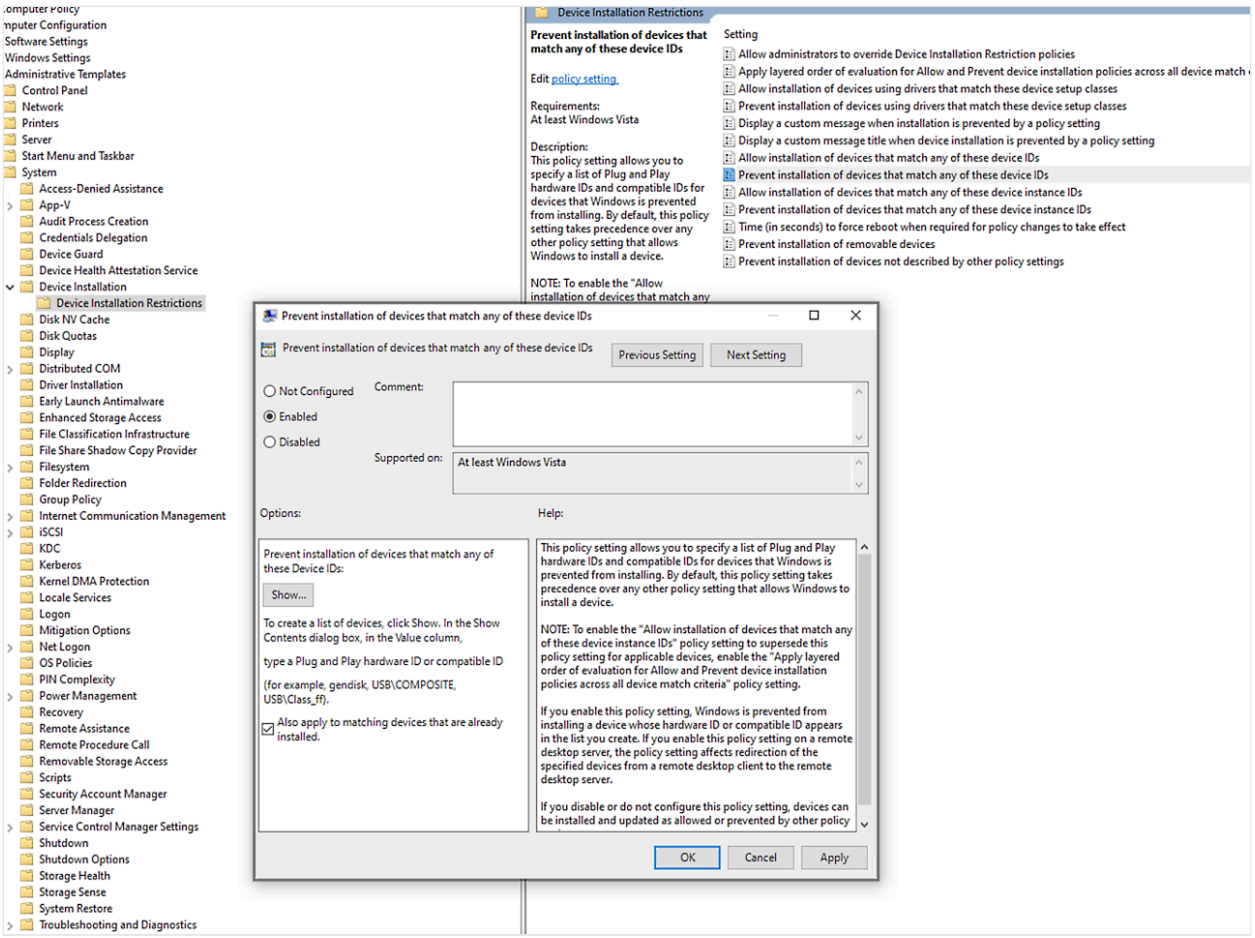
5. In Windows Explorer, browse to \\<IP address>. For example, \\169.254.76.91

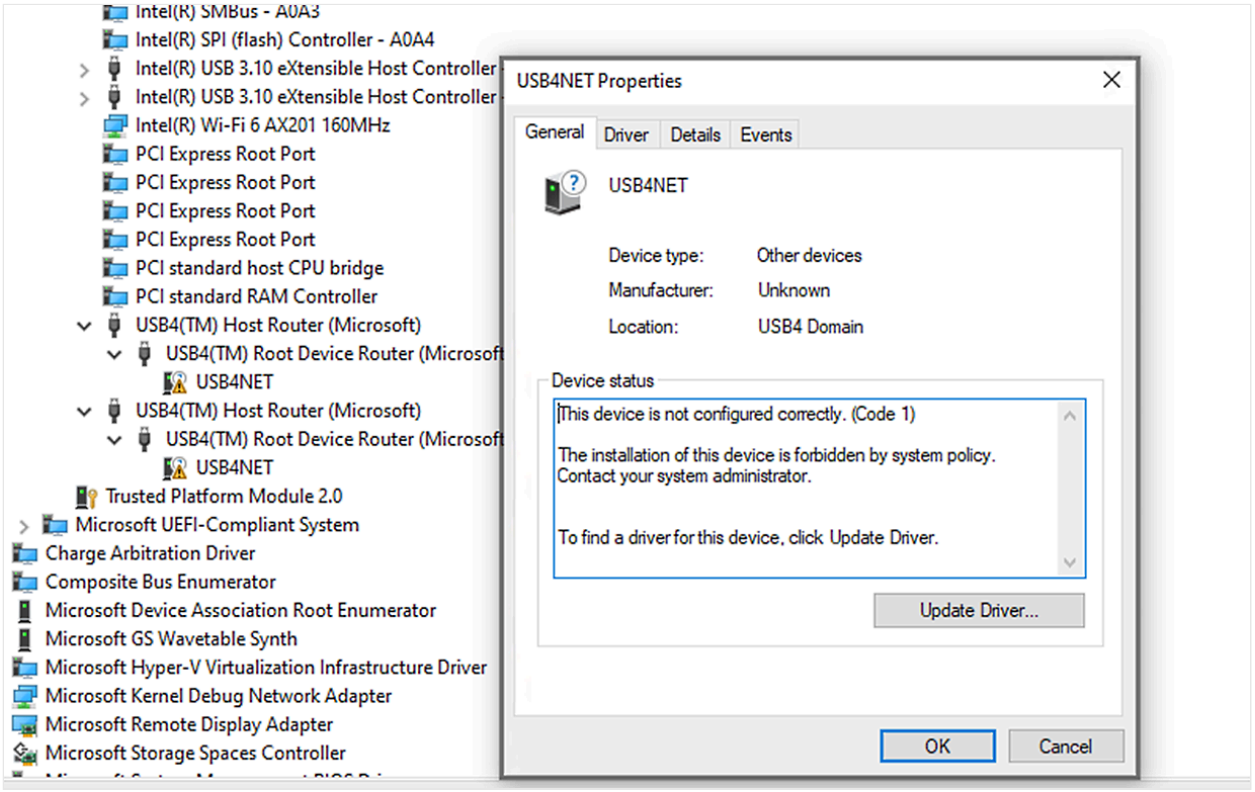
6. Sign in using an account on the remote system.

7. If the user account you used to sign in is an administrator on the remote system, you can also access admin shares. For example, \\169.254.76.91\C$

Manageability

If you would like to disable USB4 interdomain functionality, existing group policies for blocking devices based on compatible IDs can be leveraged. The Ethernet over USB4 protocol uses the compatible ID USB4\USB4NET. Blocking this compatible ID will prevent the USB4 P2P network adapter from starting.


Device Installation Restrictions
Prevent installation of devices that match any of these device IDs
Edit policy setting
Requirements:
At least Windows Vista
Description:
This policy setting allows you to specify a list of Plug and Play hardware IDs and compatible IDs for devices that Windows is prevented from installing. By default, this policy setting takes precedence over any other policy setting that allows Windows to install a device.
Setting
Allow administrators to override Device Installation Restriction policies
Apply layered order of evaluation for Allow and Prevent device installation policies across all device match
Allow installation of devices using drivers that match these device setup classes
Prevent installation of devices using drivers that match these device setup classes
Display a custom message when installation is prevented by a policy setting
Display a custom message title when device installation is prevented by a policy setting
Allow installation of devices that match any of these device IDs
Prevent installation of devices that match any of these device IDs
Allow installation of devices that match any of these device instance IDs
Prevent installation of devices that match any of these device instance IDs
Time (in seconds) to force reboot when required for policy changes to take effect
Prevent installation of removable devices
Prevent installation of devices not described by other policy settings
Prevent installation of devices that match any of these device IDs
Previous Setting
Next Setting
Not Configured
Enabled
Disabled
Comment:
Supported on:
At least Windows Vista
Options:
Help:
Show...
To create a list of devices, click Show. In the Show Contents dialog box, in the Value column,
type a Plug and Play hardware ID or compatible ID
(for example, gendisk, USB\COMPOSITE, USB\Class_ff).
Also apply to matching devices that are already installed.
OK
Cancel
Apply



The corresponding MDM policy name is "DeviceInstallation/PreventInstallationOfMatchingDeviceIDs".

Testing

Basic functionality

1. Attempt to ping the peer to confirm that the USB4 host-to-host tunnel has been set up correctly and that the network adapter has been correctly initialized.
   a. The firewall settings must be configured to allow pinging. Select **Turn on network discovery** in **Network and Sharing Center** or change the network type to **Private**, where network discovery is on by default.
2. Attempt a file transfer as described in Steps to perform a file transfer over a USB4 interdomain connection.

Throughput testing

Network throughput and stress testing can be performed using NTTTCP, which can be downloaded from GitHub.

Example test cases:

1. Testing TCP

   Run on the receiver: `NTttcp.exe -r -l 60000 -n 100M -m 8,*,169.254.X.Y -t 60`

   Run on the sender: `NTttcp.exe -s -nic 169.254.A.B -l 60000 -n 100M -m 8,*,169.254.X.Y -t 60`

   Where 169.254.X.Y is the IP address of the receiver and 169.254.A.B is the IP address of the sender.

2. Testing UDP

   Same as TCP, except add -u to the command lines.

See also

- Universal Serial Bus 4
- Introduction to the USB4 connection manager in Windows
- USB4 design details and general requirements
- USB4 ACPI requirements

- USB4 power management requirements
- USB4 required testing
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

Universal Serial Bus 4 (USB4™) required testing

Article • 12/01/2021

The HLK requirements are not designed to provide complete end-to-end functional test coverage for the USB4 feature set. We recommend that you perform thorough validation of important usage scenarios. Here are some tests that are required:

- Enable standard driver verifier on the USB4 connection manager drivers (*Usb4HostRouter.sys* and *Usb4DeviceRouter.sys*) during testing and bring-up. The drivers are designed to break into an attached kernel debugger when they detect certain violations of software or hardware requirements.

  Console

  ```
  > verifier.exe /standard /driver Usb4HostRouter.sys
  Usb4DeviceRouter.sys Wdf01000.sys
  ```

- Perform compatibility testing with various USB4 and Thunderbolt™ 3 devices and hubs.

- Perform compatibility testing with various monitors, PCIe and USB 3.x peripherals by tunneling them over a USB4 domain.

- Use the USB4 Switch to automate and stress test connect and disconnect of a USB4 or Thunderbolt™ 3 hub or device. Details are provided in How to test with the USB4 Switch.

- Validate that when the tunneled devices are idle, the USB4 domain powers down. Based on the _DSD port mappings for USB 3.x and PCIe described in the ACPI requirements for USB4, the domain should enter sleep only after the USB 3.x and PCIe devices are idle (in Dx where x > 0). For display graphics after the monitor associated with the display tunnel turns off, the USB4 domain should enter sleep.

- After the domain powers down, when a tunneled protocol device initiates wake, verify USB4 domain powers back up and the protocol wake is replayed. The tunneled devices should not undergo PnP visible re-enumeration.

- Stress test system power transitions, such as hibernate. For modern standby transitions, confirm that the USB4 domain powers down and does not block the system from entering DRIPS. For more information, see Modern Standby stress and long-duration testing.

- Across various test scenarios and transitions, ensure that the USB4 host and device routers are not in a failed PnP state. They should not display a yellow bang in device manager.
- Perform interdomain testing as described in USB4 Interdomain connections.
- Test high bandwidth monitors concurrently with USB 3.x isochronous devices like cameras. With this topology perform connect and disconnect of the peripherals and system power transitions. Ensure that the peripherals remain functional.
- Exercise DisplayPort™ alternate mode on USB4 capable ports. If there are multiple USB4 capable ports, exercise USB4 device topologies, including tunneled monitors, on one of the ports while concurrently testing DisplayPort™ alternate mode on the other ports.

How to test with the USB4 Switch

By default, the USB4 Switch is disabled in that it won't enumerate devices. To enable, download the Tools in the MUTT software package.

1. Connect the USB4 Switch to a USB4 port on the system under test.
2. Connect a Micro-B USB cable to the Switch and the other end of the Micro-B USB cable to a USB-A port on the system under test or the system you wish to use to control the switch.
3. From an elevated command window, navigate to where the MUTT tools are located. Run this to confirm that the USB4 Switch is detected:

   Console

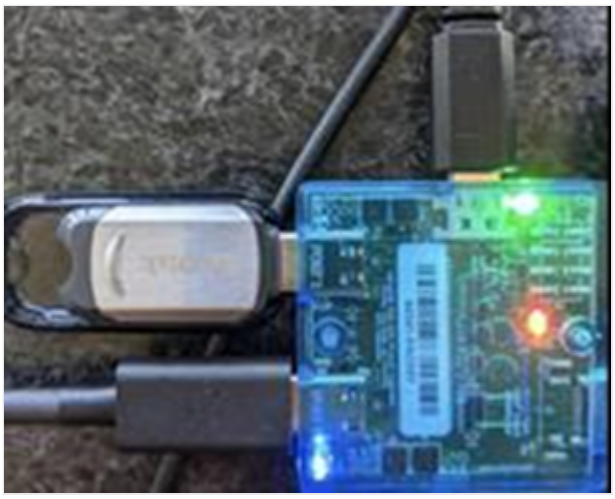
   ```
   > ConnExUtil.exe /list
   1 Connection Exerciser Device Attached
   \\?\usb#vid_045e&pid_0646&mi_00#6&1456cc2b&0&0000#{86e0d1e0-8089-11d0-
   9ce4-08003e301f73}
   ```

4. Run this to switch to Port 1:

   Console

   ```
   > ConnExUtil.exe /setport 1
   1 ConnectiCon Exerciser Device Attached
   Switching to Port: 1
   ```

Once you run this, it will enumerate whatever is connected to port 1 and you should see a blue LED light up, as shown in this picture:



> **Note**
>
> If the LED doesn't light up, the switch may still be functioning properly. You can confirm this by checking that the downstream device is successfully enumerated in the device manager or using "connexutil /getPort".

5. Run the following command to switch to Port 2:

```
Console

> ConnExUtil.exe /setport 2
1 Connection Exerciser Device Attached
Switching to Port: 2
```

6. Another test that is of very high value is to run /CxStress for stress testing.

```
Console

> ConnExUtil.exe /cxstress
```

7. In addition to the test above, stress tests can be run while varying the time between connect and disconnect. This can be helpful to give enough time for the USB4 domain to power down before connect or disconnect.

```
Console

> ConnExUtil.exe /cxstress /mintime 5000 /maxtimeconnected 10000
/maxtimedisconnected 10000
```

See also

- Universal Serial Bus 4
- Introduction to the USB4 connection manager in Windows
- USB4 design details and general requirements
- USB4 ACPI requirements
- USB4 power management requirements
- USB4 interdomain connections
- USB4 debugging and troubleshooting

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

Universal Serial Bus 4 (USB4™) debugging and troubleshooting

Article • 10/25/2022

This section provides guidelines for debugging and troubleshooting problems with Universal Serial Bus 4 (USB4™) device drivers. See the following articles for more information on USB4™ debugging and troubleshooting.

- Examining connection manager driver traces
- TraceLogging rundown events
- USB4 kernel debugger extension
- System event log events for display tunnels
- USB4 in the Windows device portal

See also

- Universal Serial Bus 4
- Introduction to the USB4 connection manager in Windows
- USB4 design details and general requirements
- USB4 ACPI requirements
- USB4 power management requirements
- USB4 interdomain connections
- USB4 required testing

"USB4™" is a trademark of USB Implementers Forum and is only intended for use with products based on and compliant with the USB4™ specification.

Examining connection manager driver traces for USB4™ devices

Article • 12/08/2022

This article explores how to capture and review connection manager driver traces for USB4 devices.

Capture traces into ETL files

The *BusesTrace.cmd* script can be used to capture the traces of USB4 connection manager drivers and other USB drivers into ETL files. Tools such as Windows Performance Analyzer can be used to examine the traces in these ETL files.

The following steps capture the traces from the current boot session.

1. Launch the *BusesTrace.cmd* script from an elevated command prompt.
2. Choose the script menu items **Start Tracing**, **USB4 components** and **Start Now** to start the trace session.
3. <Reproduce the problem>
4. Follow the *BusesTrace.cmd* script prompt to hit any key to stop the trace session.

The following steps capture the traces from the beginning of the next boot session.

1. Launch the *BusesTrace.cmd* script from an elevated command prompt.
2. Choose the script menu items **Start Tracing**, **USB4 components** and **Start From Next Boot Session** to configure the trace session for next boot sessions.
3. Reboot the system. The trace session starts after the reboot.
4. <Reproduce the problem>
5. Launch the *BusesTrace.cmd* script again from an elevated command prompt.
6. Choose the script menu item **Stop Boot Session Trace** to stop the boot trace session and remove the boot trace session configuration for next boot sessions.

Examine traces in IFR circular memory buffer

The USB4 connection manager drivers implement Inflight Trace Recorder (IFR). This allows for the WPP trace messages to be written to a circular memory buffer and is always enabled. You do not need to enable a trace session before-hand. The traces messages from the IFR are accessible from a kernel debugger or kernel memory dump.

Use the kernel debugger commands to examine the traces.

Console

```
!rcdrkd.rcdrlogdump Usb4HostRouter.sys
!rcdrkd.rcdrlogdump Usb4DeviceRouter.sys
```

TraceLogging USB4™ rundown events

Article • 10/25/2022

The USB4 host router and device router drivers emit TraceLogging events to report the state of the domain. These events are known as rundown events and are reported by the drivers when a ETW trace session for the following trace providers is enabled. These events describe the domain through a series of events. After the rundown events are emitted, the USB4 drivers will continue to report events on router enumeration and removal.

ETW provider name: "Microsoft.Windows.USB.USB4.HostRouter" ETW provider GUID: {575BA31F-2B45-58C2-64FD-F5DC757B6137}

ETW provider name: "Microsoft.Windows.USB.USB4.DeviceRouter" ETW provider GUID: {AE795D36-2B11-5EFB-C7E0-5D552BC55D6C}

The following tables describe some TraceLogging events of interest.

1. Event **RundownStart** and **RundownComplete** are marker events to denote start and end of rundown events. They do not carry any event properties or payload with them.

2. Event **DeviceRouterInformation**

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event. False if it is reporting a new device router.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of this device router.
DeviceInstancePath	WideString	The PnP-assigned device instance path of the device router. It can be used to match the DEVPKEY_Device_InstanceId property data that PnP APIs in the Unified Device Property Model return.
VendorId	HexUInt16	Vendor ID
ProductId	HexUInt16	Product ID

Property	Value type	Description
AsciiVendorName	String	The vendor name in ASCII. It is defined in the ASCII Vendor Name Entry chapter of the USB4 DROM specification.
AsciiModelName	String	The model name in ASCII. It is defined in the ASCII Model Name Entry chapter of the USB4 DROM specification.
DeviceID	HexUInt16	For Thunderbolt™ 3, it's the "VendorId" field in the Thunderbolt™ 3 header section. For USB4, it's the "idVendor" field in the Product Descriptor Entry.
ModelID	HexUInt16	For Thunderbolt™ 3, it's the "Model ID" field in the Thunderbolt™ 3 header section. For USB4, it's the "idProduct" field in the Product Descriptor Entry.
UUID	HexUInt64	The device UUID. For USB4, it is the UUID in the ROUTER_CS_7 and ROUTER_CS_8 registers defined by the USB4 specification.
DeviceFirmwareVersion	HexUInt32	The device firmware version.

3. Event **PortInformation**

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event. False if it is reporting a new device router.
IsNewDeviceRouter	Boolean	True if it is reporting a port of a new device router. False if it is reporting a port status change. For rundown events, it is also false.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of this device router.
IsDFP	Boolean	True if the associated Port is DFP. False if it's UFP.
Lane0AdapterNumber	UInt8	The "Adapter Number" from adapter configuration space register ADP_CS_3.

Property	Value type	Description
Lane1AdapterNumber	UInt8	The "Adapter Number" from adapter configuration space register ADP_CS_3.
DownstreamRouterDomainID	HexUInt32	The domain ID of the downstream device router.
DownstreamRouterTopologyID	UInt8FixedArray(7)	The 7-byte topology ID of the downstream device router. All zeros if no device router is connected downstream.
SupportedLinkSpeeds	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_0[19..16] "Supported Link Speeds".
SupportedLinkWidths	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_0[25..20] "Supported Link Widths".
CurrentLinkSpeed	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_1[19..16] "Current Link Speed".
NegotiatedLinkWidth	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_1[25..20] "Negotiated Link Width".
TargetLinkSpeed	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_1[3..0] "Target Link Speed".
TargetLinkWidth	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_1[9..4] "Target Link Width".
AdapterState	HexUInt8	The value from lane adapter configuration capability register LANE_ADP_CS_1[29..26] "Adapter State".
LaneBonded	Boolean	Indicates whether the lanes of this port are bonded.

Property	Value type	Description
CableUsbVersion	Boolean	The value from port capability register PORT_CS_18[7..0] "Cable USB Version".
LinkCLxSupport	Boolean	The value from port capability register PORT_CS_18[10] "Link CLx Support (LCL)".
Tbt3CompatibleMode	Boolean	The value from port capability register PORT_CS_18[9] "TBT3-Compatible Mode (TCM)".

4. Event **PCIeAdapterInformation**

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event. False if it is reporting a new device router.
IsNewDeviceRouter	Boolean	True if it is reporting a port of a new device router. False if it is reporting a port status change. For rundown events, it is also false.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of this device router.
AdapterNumber	Uint8	The "Adapter Number" from adapter configuration space register ADP_CS_3.
AdapterType	HexUInt32	Adapter configuration space register ADP_CS_2[23..0]. It combines the values of "Adapter Type Protocol", "Adapter Type Version" and "Adapter Type Sub-Type". Spec reference: "Table 8-9. Adapter Configuration Space Basic Attributes" and "Table 8-10. Adapter Types"
IsDownstream	Boolean	True if it's a downstream PCIe adapter. False if it's upstream.
IsTunneled	Boolean	True if the protocol traffic is tunneled.

5. Event **DPAdapterInformation**

Property	Value type	Description

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event. False if it is reporting a new device router.
IsNewDeviceRouter	Boolean	True if it is reporting a port of a new device router. False if it is reporting a port status change. For rundown events, it is also false.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	Uint8FixedArray(7)	The 7-byte topology ID of this device router.
AdapterNumber	Uint8	The "Adapter Number" from adapter configuration space register ADP_CS_3.
AdapterType	HexUInt32	Adapter configuration space register ADP_CS_2[23..0]. It combines the values of "Adapter Type Protocol", "Adapter Type Version" and "Adapter Type Sub-Type". Spec reference: "Table 8-9. Adapter Configuration Space Basic Attributes" and "Table 8-10. Adapter Types"
IsDPOut	Boolean	True if it is a DP Out adapter. False if it is DP In.
IsTunneled	Boolean	True if the protocol traffic is tunneled.
MaximalLinkRate	HexUInt8	The value from DP adapter configuration space register DP_COMMON_CAP[11..8] "Maximal Link Rate".
MaximalLinkCount	HexUInt8	The value from DP adapter configuration space register DP_COMMON_CAP[14..12] "Maximal Link Count".

6. Event **USB3AdapterInformation**

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event. False if it is reporting a new device router.
IsNewDeviceRouter	Boolean	True if it is reporting a port of a new device router. False if it is reporting a port status change. For rundown events, it is also false.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of this device router.

Property	Value type	Description
AdapterNumber	UInt8	The "Adapter Number" from adapter configuration space register ADP_CS_3.
AdapterType	HexUInt32	Adapter configuration space register ADP_CS_2[23..0]. It combines the values of "Adapter Type Protocol", "Adapter Type Version" and "Adapter Type Sub-Type". Spec reference: "Table 8-9. Adapter Configuration Space Basic Attributes" and "Table 8-10. Adapter Types"
IsDownstream	Boolean	True if it is a downstream USB 3.x adapter. False if it is upstream.
IsTunneled	Boolean	True if the protocol traffic is tunneled.
ActualLinkRate	HexUInt8	The value from USB 3.x adapter configuration space register ADP_USB3_CS_4[6..0] "Actual Link Rate".

7. Event **OtherAdapterInformation**

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event. False if it is reporting a new device router.
IsNewDeviceRouter	Boolean	True if it is reporting a port of a new device router. False if it is reporting a port status change. For rundown events, it is also false.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of this device router.
AdapterNumber	UInt8	The "Adapter Number" from adapter configuration space register ADP_CS_3.
AdapterType	HexUInt32	Adapter configuration space register ADP_CS_2[23..0]. It combines the values of "Adapter Type Protocol", "Adapter Type Version" and "Adapter Type Sub-Type". Spec reference: "Table 8-9. Adapter Configuration Space Basic Attributes" and "Table 8-10. Adapter Types"

8. Event **InterDomainPeerInformation**

Property	Value type	Description
IsRundownEvent	Boolean	True if it is a rundown event.
DomainID	HexUInt32	The domain ID of this device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of this device router.
Lane0AdapterNumber	UInt8	The lane 0 adapter number of the local port. Same as Lane0AdapterNumber of the "PortInformation" event. It can be used together with the "DomainID" and "TopologyID" properties above to find the matching "PortInformation" event.
LocalDomainUUID	GUID	The domain UUID of the local domain in an inter-domain connection.
RemoteDomainUUID	GUID	The domain UUID of the remote domain in an inter-domain connection.
LocalRouteString	UInt8FixedArray(7)	The route string for the local port. It's in the 7-UINT8 topology ID format.
RemoteRouteString	UInt8FixedArray(7)	The route string for the remote port. It's in the 7-UINT8 topology ID format
MaxHopID	UInt16	The "MaxHopID" property from the inter-domain properties block returned by the remote peer.

9. Event **DeviceRouterRemoval**

Property	Value type	Description
DomainID	HexUInt32	The domain ID of the device router.
TopologyID	UInt8FixedArray(7)	The 7-byte topology ID of the device router.

Thunderbolt is a trademark of Intel Corporation or its subsidiaries.

Universal Serial Bus 4 (USB4™) kernel debugger extension

Article • 10/25/2022

To assist with debugging from a kernel debugger, the following *USB4KD.dll* is part of the Windows Debugger Tools starting with the Windows SDK release for Windows 11. The latest version of WinDbg can be obtained from WinDbg Preview - Microsoft Store

The following commands are available. To begin, examine the USB4 domains on the system with `!usb4kd.usb4tree` and follow the links to examine the routers, ports, adapters and tunnels.

Console

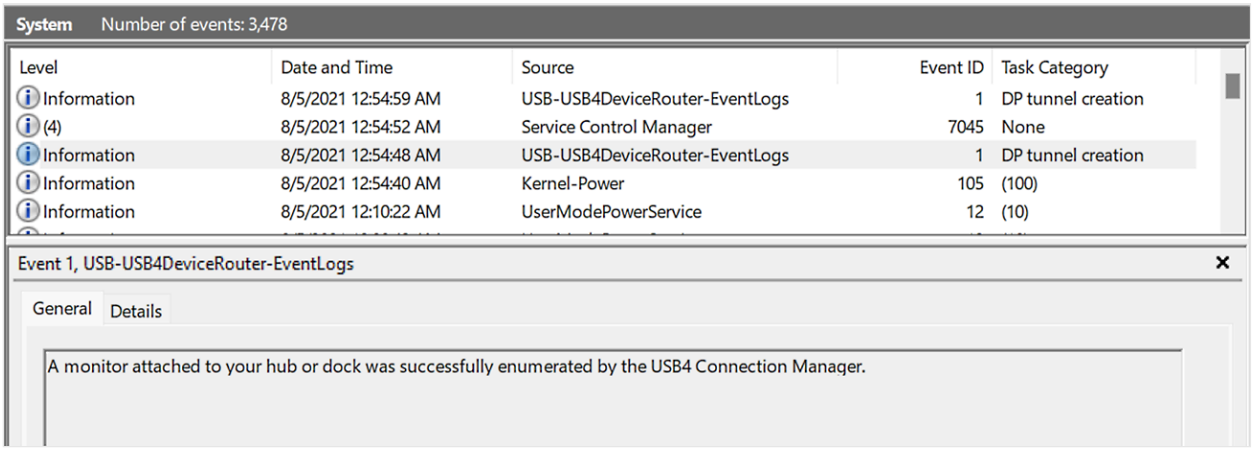
```
0: kd> !usb4kd.help
Commands for usb4kd.dll:
  !help              - Displays information on available extension commands
  !usb4adapter       - Dumps information about the USB4 Adapter
  !usb4devicerouter  - Dumps information about the USB4 Device Router
  !usb4help          - Displays information on available extension commands.
                       Alias to !usb4kd.help
  !usb4hostrouter    - Dumps information about the USB4 Host Router
  !usb4port          - Dumps information about the USB4 Port
  !usb4rbcsregisters - Dumps registers from the USB4 Router Basic
Configuration
                       Space
  !usb4tree          - Dumps USB4 hierarchy of routers and ports
  !usb4tunnel        - Dumps information about the USB4 Tunnel
  !usb4tunnels       - Dumps information about all USB4 Tunnels
!help <cmd> will give more information for a particular command
```

System event log events for display tunnels

Article • 10/25/2022

The connection manager writes display tunnel creation and failure events to the system event log to assist with troubleshooting.

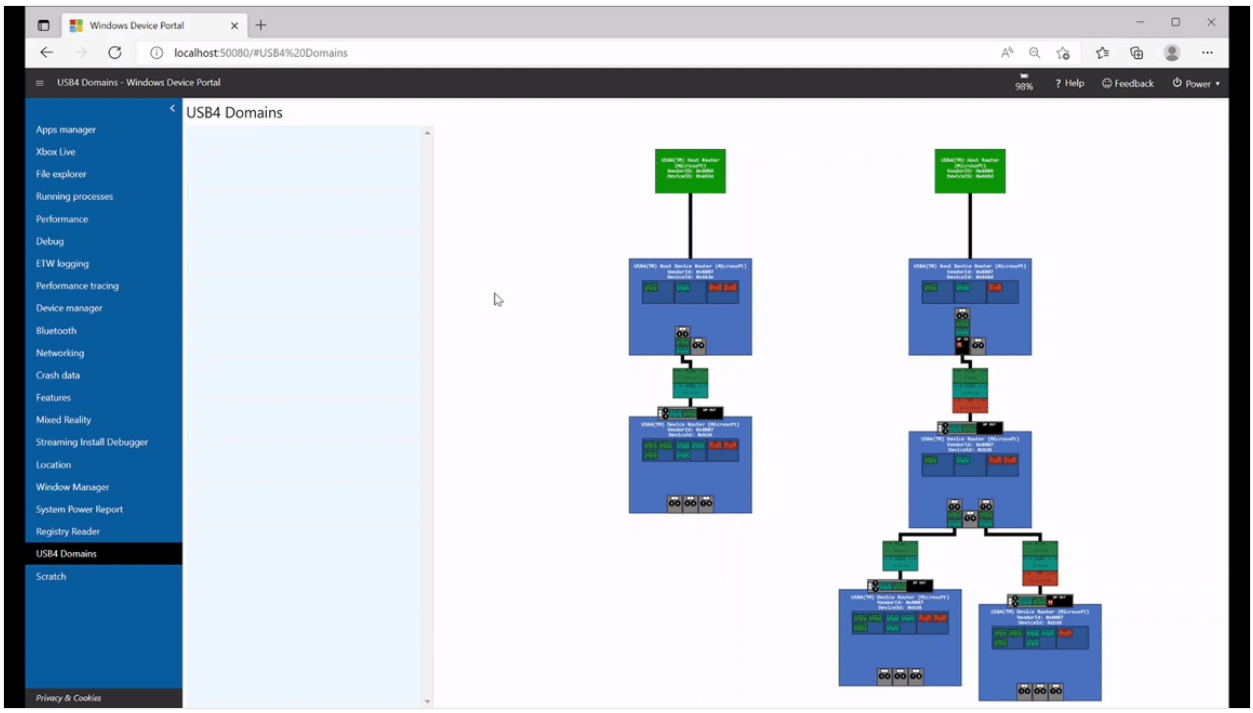
Event ID	Source	Event Message	OpCode
1	USB-USB4DeviceRouter-EventLogs	"A monitor attached to your hub or dock was successfully enumerated by the USB4 Connection Manager."	win:Info
2	USB-USB4DeviceRouter-EventLogs	"A monitor attached to your hub or dock could not be enumerated by the USB4 Connection Manager. This may be expected if the dock or system does not have sufficient resources or bandwidth to support the monitor. ReasonCode1:%13 ReasonCode2:%14 Status:%15"	win:Error



Universal Serial Bus 4 (USB4™) in the Windows device portal

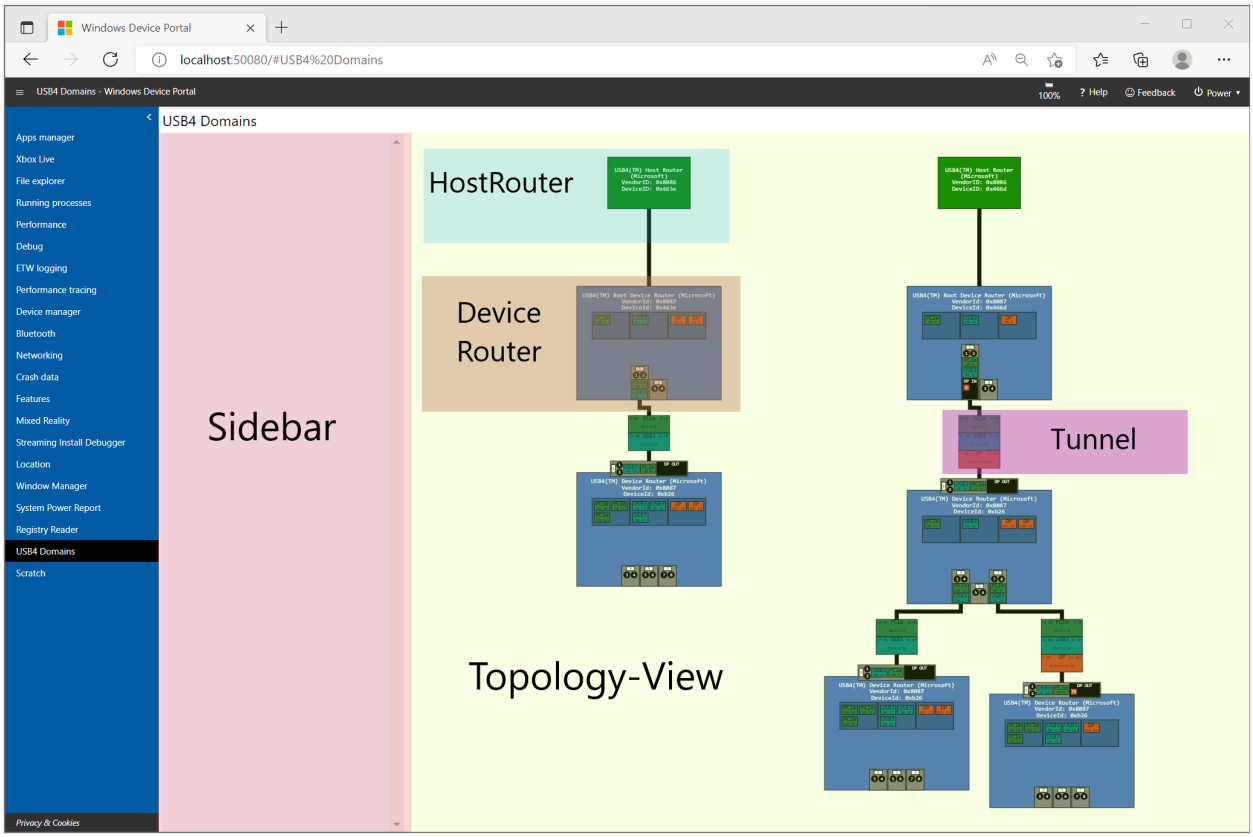
Article • 10/25/2022

Demonstration



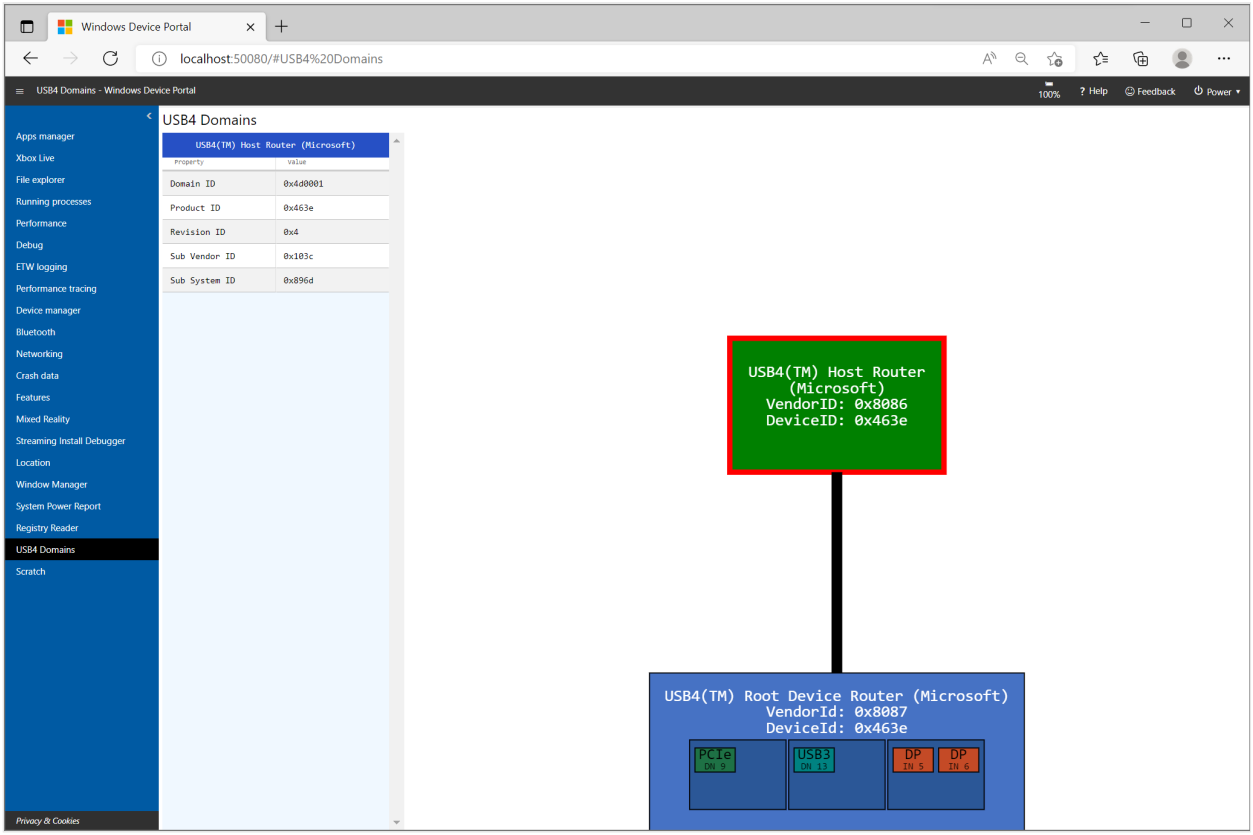
Topology

- The USB4 domain viewer is accessed via the Windows device portal.
- USB4 topology view shows USB4 host routers and downstream connected USB4 device routers.
- When elements are selected, the sidebar is updated with element-specific data.
- Topology view can be navigated with click and drag.
- Topology view can be zoomed in and out with the mouse scrollwheel.
- Topology view is automatically updated as the topology changes.



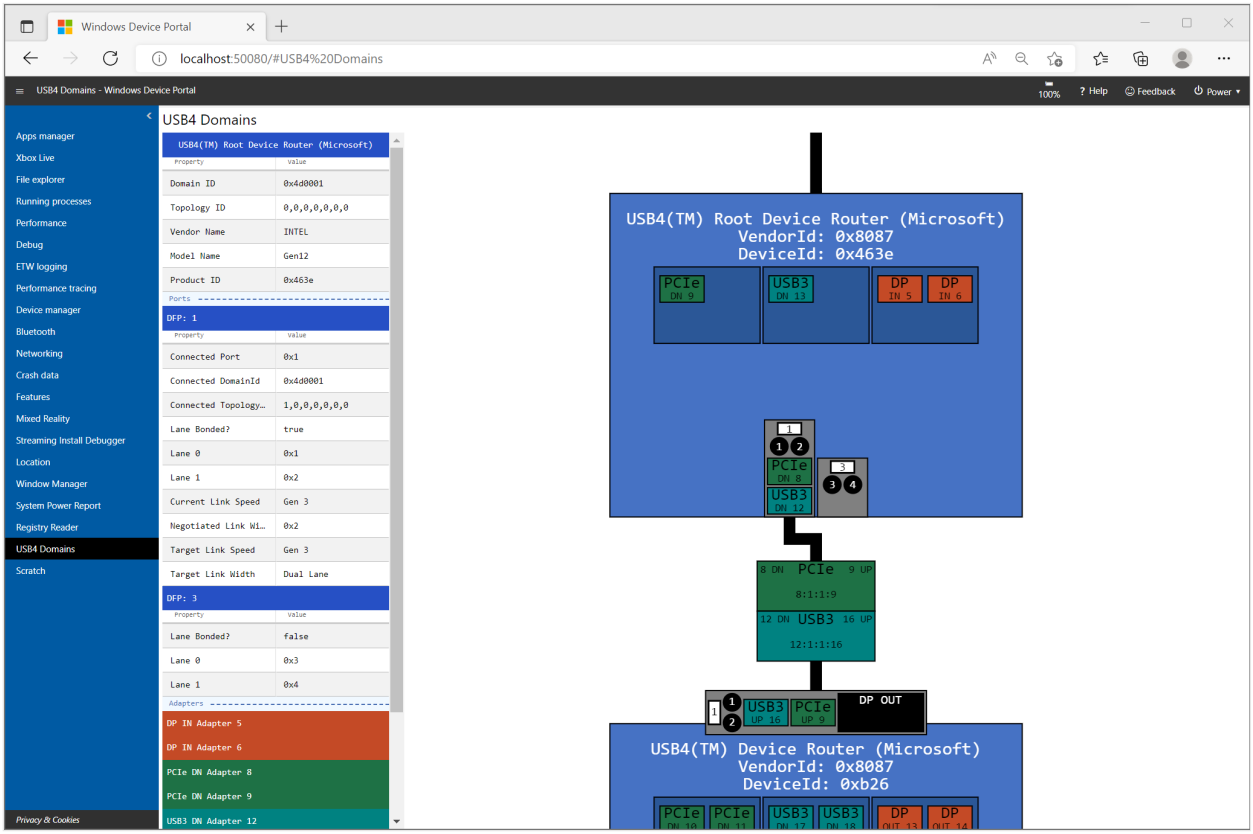
Host router

- Host routers are illustrated in green and contain a brief description including Name, VendorId, and DeviceId.
- Selecting a host router updates the sidebar with additional details.



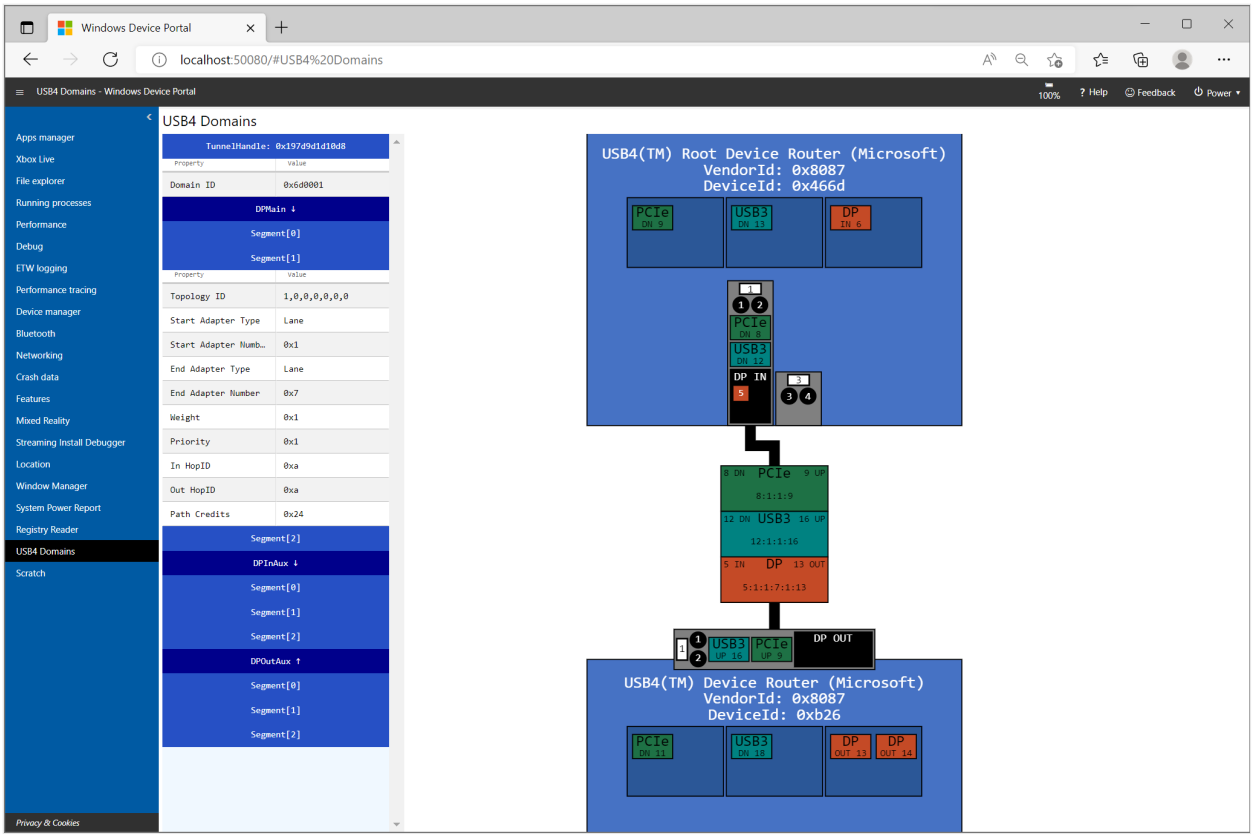
Device router

- Device routers are illustrated in blue and contain a brief description along with adapters and ports.
- In-use adapters are placed within the associated port. Unused adapters are still visible but are separated by type and not placed within a port.
- Selecting a device router updates the sidebar with additional details.
- Selecting an adapter from the device router sidebar highlights the adapter with a red circle.



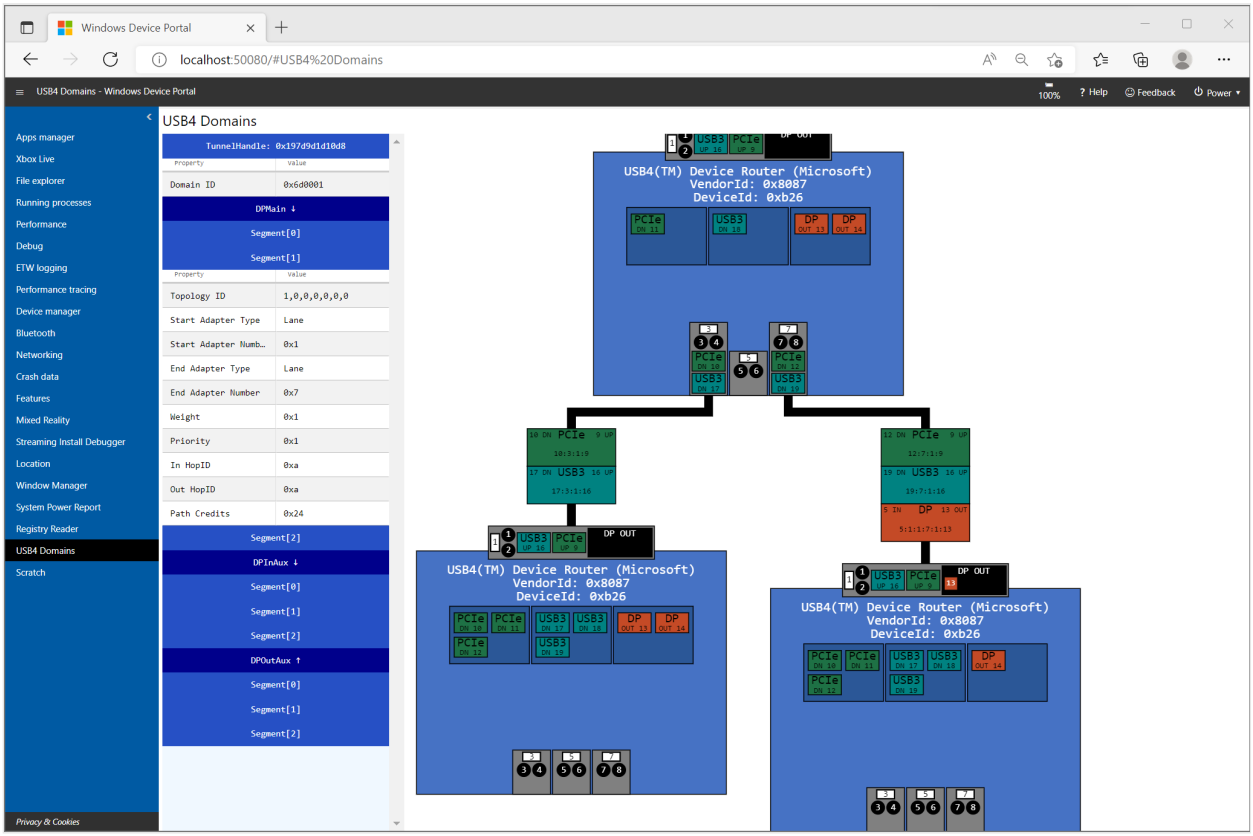
Tunnels

- Tunnels are illustrated between connected device routers. If the tunnel travels between multiple device routers it will be illustrated multiple times, once in each connection.
- Selecting a tunnel updates the sidebar with additional details.
- Each tunnel describes its path at a glance by combining adapter IDs into a colon separated unique ID. For example, 8:1:1:9.



Sidebar

- Sidebar data is context specific. Selecting different items in the diagram updates the sidebar.
- All sidebar information can be collapsed or expanded by clicking the box above the table.



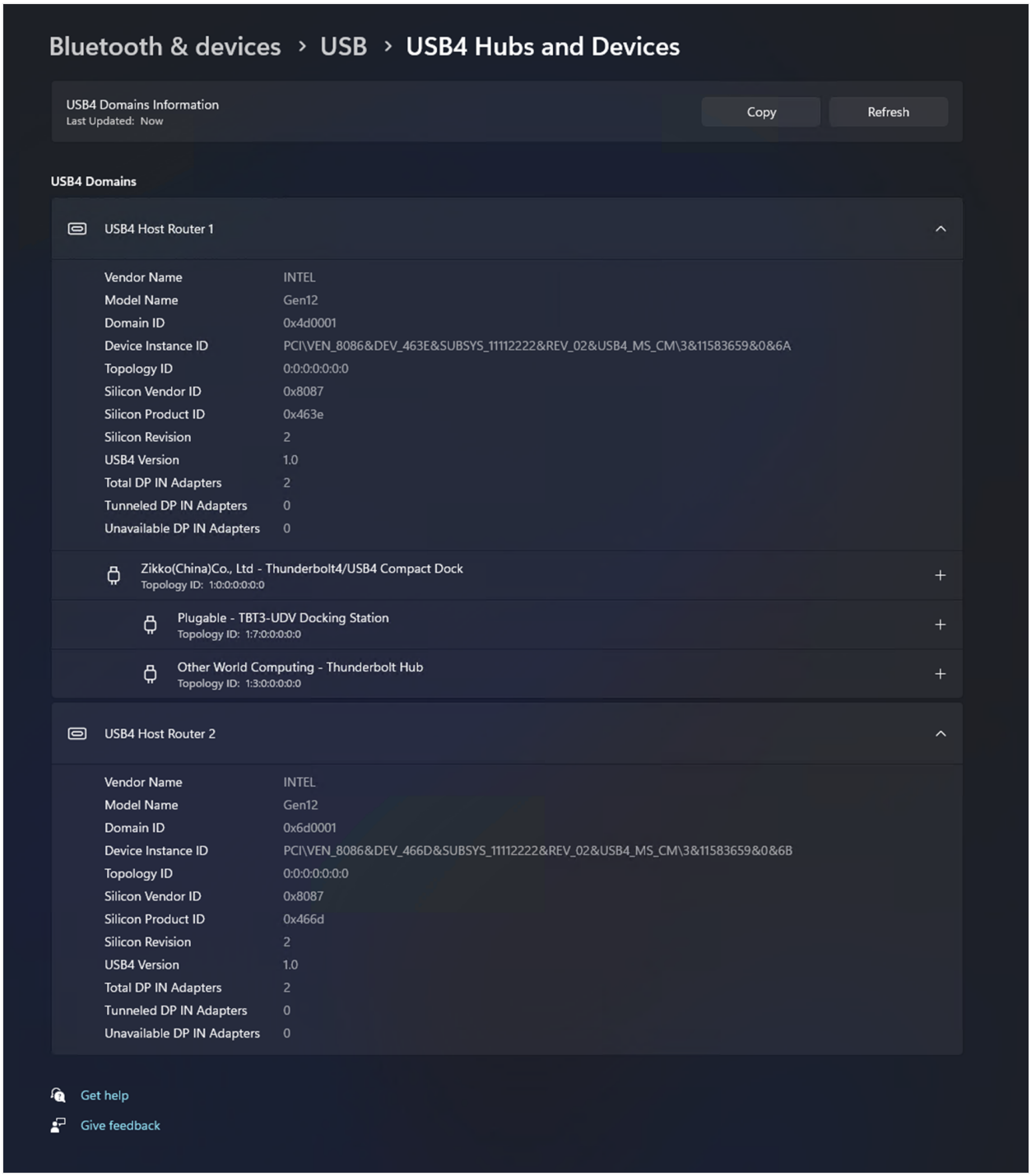
Universal Serial Bus 4 (USB4™) settings enablement

Article • 03/01/2024

This article explains how to enable Universal Serial Bus 4 (USB4™) settings in Windows.

Starting in Windows 11 build 22621.1778 (KB5026446), the Windows Settings app lists attached USB4 hubs and devices and their capabilities. For USB4-capable systems, navigate in the Settings app to **Bluetooth & devices** > **USB** > **USB4 Hubs and Devices**.

The **USB4 Hubs and Devices** page is enabled when the presence of any USB4 host router is detected.



> **Note**
>
> If a system is not USB4-capable, the **USB4 Hubs and Devices** page will not appear.

If the USB4 host router is statically or always enumerated, then the USB4 settings page will appear automatically. This is true for most PCI enumerated USB4 host routers. If the USB4 host router is dynamically enumerated only when a USB4 hub or device is attached, Microsoft recommends that OEMs set the following registry value. If the *IsSystemUsb4CapableFromOem* registry value is set, the USB4 settings page will be visible even when a USB4 hub or device has not been connected.

Output

```
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\USB]
"IsSystemUsb4CapableFromOem"=dword:00000001
```

Windows Hardware Design Whitepapers

Article • 01/18/2019

For each device class in the Windows platform, hardware designers and driver developers require unique implementation tips and tools for ensuring reliability and good performance. You can find information here about issues that are important for designers and driver developers who create products that run on systems running Windows.

Device Profile For Web Services Location Extension

Article • 06/21/2022

Last updated:

- April 29, 2015

Applies to:

- Windows 10

Information about the Device Profile for Web Services Location Extension.

File name: DeviceLocation.zip

39.1 KB

Contains a Microsoft Word document and supporting file.

Get Office File Viewers

Download

Included in this paper:

- Information about the device location namespace
- A description of the DeviceLocation element and associated values
- A description of the device location XSD schema

HID Sensors Usages

Article • 06/21/2022

Last updated:

- November 5, 2015

Applies to:

- Windows 10
- Windows 8

This paper provides information about the HID Sensor Class Driver for Windows 8 and later operating systems. It provides guidelines for developing sensor hardware and firmware that take full advantage of, and work correctly with the in-box driver.

File name: hid-sensors-usages.docx

490 KB

Microsoft Word file

Get Office File Viewers

Download

Included in this white paper:

- Sensor Page (0x20)
- Sensor Backgrounder
- Sensor Interaction via HID
- Illustrative Examples
- Sensor Implementation and Debugging – Tips & Tricks

Implementing Web Services on Devices for Printing

Article • 12/15/2020

Last updated:

- April 29, 2015

Applies to:

- Windows Vista
- Windows 7
- Windows 8
- Windows 8.1
- Windows 10

Information about web services that provide a connection protocol for printing and scanning peripherals. The Web Services technology provides a common framework for describing and sharing information.

Windows Vista was the first Microsoft Windows operating system to provide Web Services on devices as a connection protocol for printing and scanning peripherals. Web Services technology provides a common framework for describing and sharing information. Web Services on devices, part of the Windows Rally program, describes a set of protocols for consuming and controlling services on network-connected devices.

Four Web Services specifications exist for printing and scanning, to help device manufacturers take advantage of the improved customer experience around connecting, installing, and using devices with Windows.

For Windows 8.1, the print service definition for web services on devices has been updated to v1.2.

For Windows 10, the print service definition for web services on devices has been updated to v2.0.

Highlights in WS-Print v2.0

WS-Print V2.0 adds a number of new schema elements and operations that supplements the current WS-Print V1.0 service specification. The basic operation of a WS-Print V1.0 print device has not changed. The new operations and schema elements

in WS-Print V2.0 allow a mobile client to print without the aid of a driver, and can be used to enhance the capabilities of a v4 print driver.

WS-Print v2.0 defines the following new operations:

- **PrepareToPrint** – Informational operation which allows printers to warm up
- **CreatePrintJob2** – Extension of CreatePrintJob which includes PrintSchema PrintTicket for job configuration
- **GetPrintDeviceCapabilities** – Allows retrieval of a PrintDeviceCapabilities document
- **GetPrintDeviceResources** – Allows retrieval of localized resources in ResX
- **GetBidiSchemaExtensions** – Allows retrieval of Bidi Schema extensions

For more information, see Appendix E in the Print Service Definition v2.0 specification available in the WS-Print v2.0 file download below.

Highlights in WS-Print v1.2

Web Services on Devices for printing (WS-Print) v1.2 adds a number of new schema elements to the current WS-Print V1.1 Service Specification.

WS-Print includes all operations and schema elements used in WS-Print V1.1, but adds support for a new schema element and a new operation. The new schema element "SupportsWSPrintV12" is used to identify support for WS-Print V1.2. The new operation, "SetPrinterElements" enables a client to set the value of a schema element on the printer. For example, the client could set a custom element called "InkHeadAlignmentValue" which the printer would use to realign the inkjet head.

In order to facilitate specification implementation and understanding, the specifications are also available here in complete, stand-alone form, along with their associated Web Services Description Languages (WSDLs) and XML Schema Definitions (XSDs). These four Web Services on devices specifications are covered by the included technical documentation license agreement, which references the WDK.

File downloads

Download the Specification and Supporting Files for WS-Print v2.0

Print Service Definition v2.0 for Web Services on Devices

(1.55 MB zip file containing Microsoft Word document and supporting files; April 29, 2015)

Download the Specification and Supporting Files for WS-Print v1.2

Print Service Definition v1.2 for Web Services on Devices

(2.64 MB zip file containing Microsoft Word document and supporting files; September 16, 2013)

Download the Specifications and Supporting Files

Print Device Definition V1.0 for Web Services on Devices

(38 KB zip file containing Microsoft Word document and supporting file; January 29, 2007)

Scan Service Definition V1.0 for Web Services on Devices

(1.5 MB zip file containing Microsoft Word document and supporting files; February 9, 2012)

Scan Device Definition V1.0 for Web Services on Devices

(38 KB zip file containing Microsoft Word document and supporting file; January 29, 2007)

Integrating Ambient Light Sensors with Computers Running Windows 10 Creators Update

Article • 10/20/2022

Last updated:

- Mar 25, 2019

Applies to:

- Windows 10
- Windows 11

This paper provides information about ambient light sensor (ALS) and adaptive brightness features in the Windows 10 operating system. It provides guidelines for hardware manufacturers and original equipment manufacturers (OEMs) to integrate ALS hardware with computers that use the adaptive brightness feature and with applications that use light sensor data.

File name: integrating-ambient-light-sensors-with-windows-10.docx

0.98MB

Microsoft Word file

Download

Included in this white paper:

- Light Sensor Features
- Ambient Light Sensor Integration Checklist
- Ambient Light Sensor Calibration
- Understanding Adaptive Brightness Support in Windows 10 Creator's Update
- Types of Light Sensors
- Integrating Light Sensors with Device Hardware
- Optimizing Display Brightness Steps and Transitions
- Tuning and Configuring Adaptive Brightness Behavior
- Defining a Custom Ambient Light Response (ALR) Curve in the Registry

Integrating Motion and Orientation Sensors

Article • 06/21/2022

Last updated:

- February 28, 2018

Applies to:

- Windows 10 and earlier

Motion and orientation sensor capabilities and requirements for Windows 10 and earlier operating systems.

File name: integrating-motion-and-orientation-sensors.docx

10.9 MB

Microsoft Word file

Get Office File Viewers

Download

This paper is intended to help OEMs, ODMs, and IHVs understand motion and orientation sensor capabilities and requirements for Windows 10 and earlier operating systems. The paper also shows how these capabilities are engineered and implemented on PC hardware platforms running Windows. This technology area involves 3D accelerometer, 3D magnetometer/compass, and 3D gyrometer sensor hardware.

This paper does not cover application development or sensor technology that is not directly related to orientation and motion detection.

Included in this white paper:

- Overview of motion and device
- Orientation implementation guidelines
- Calibration

V4 print driver improvements

Article • 06/21/2022

Last updated:

- April 29, 2015

Applies to:

- Windows 10

Information about v4 print driver improvements in Windows 10.

File name: V4_Print_Driver_Improvements.docx

30.5 KB

Microsoft Word file

Get Office File Viewers

Download

Included in this paper:

In order to support WS-Print v2.0 devices, PWG Raster rendering, PrintDeviceCapabilities files and .resx files, Windows 10 will provide a number of improvements which are available for all v4 print drivers running on Windows 10. This article contains information about the following items:

- GPD files
- A PrintSchemaNamespace example
- PWG Raster standard rendering filter and pipeline configuration file
- Requirements for the DriverConfig and DriverRender sections of the v4 print driver manifest

Windows Keyboard Layouts

Article • 06/21/2022

Last updated:

- March 17, 2015

Applies to:

- Windows 10
- Windows 8
- Windows 7

Overview of keyboard layouts and gives guidelines for developing PS/2-based and human interface device (HID)-based keyboards.

File name: windows-keyboard-layouts.docx

104 KB

Microsoft Word file

Get Office File Viewers

Download

This paper helps OEMs, ODMs, and IHVs develop keyboards with different layouts that work effectively with Windows. It provides a brief overview of keyboard layouts and gives guidelines for developing PS/2-based and human interface device (HID)-based keyboards. It also includes two samples that enable partners to easily deploy the solution.

Included in this white paper:

- Overview of keyboard layouts on Windows driver guidance
- Sample INF

Windows Hardware Compatibility Program

Article • 06/24/2021

The Windows Hardware Compatibility Program is designed to help your company deliver systems, software and hardware products that are compatible with Windows and run reliably on Windows 10, Windows 11 and Windows Server 2022. The program also provides guidance for developing, testing and distributing drivers. Using the Windows Hardware Dev Center dashboard, you can manage submissions, track the performance of your device or app, review telemetry and much more.

Windows Hardware Lab Kit

The Windows Hardware Compatibility Program leverages the tests in the Windows Hardware Lab Kit, a test framework used to test hardware devices for Windows 10 and Windows 11.

Playlists

The Windows Hardware Compatibility Program uses an official playlist to determine which devices meet the requirements for compatibility with Windows 10 and Windows 11. All playlists that we have published are acceptable to use for submissions to the Windows Hardware Dev Center.

The latest playlist can be downloaded at: https://aka.ms/HLKPlaylist

Blog

For the most up-to-date news about the program, check out the Windows Hardware Certification Blog.

Customize the Windows device UI

Article • 06/07/2023

Purpose

Customizations of the Windows OS are ways in which partners can modify the Windows device UI, connectivity settings, and user experience to better reflect the partners' brand, and to fit the network and market in which the device ships. Customization options include adding applications, modifying icons and Start layouts, configuring network settings by using device management, changing defaults in **Settings**, and adding brand-specific art and sounds to the OS.

See the following sections for more information about what you can do to customize your devices.

Expand table

Topic	Description
Customizations for desktop	This section includes topics describing key desktop customization opportunities, as well the Unattended Windows Setup Reference, and Windows System Image Manager Technical Reference.
Customizations for enterprise desktop	Learn about the customizations available to you if you are providing a controlled and specialized experience on a Windows device running Windows 10 Enterprise.
Configure power settings	Learn about the power settings you can configure using the Windows provisioning framework. Each power setting topic includes the identification GUID, allowed values, meaning, and common usage scenarios for the setting.
Preinstalled and exclusive apps	If you're a Windows OEM or mobile operator partner, find out how you can create partner apps that you can package and configure to install during the initial device setup process. While the user is going through the initial setup process, the preinstalled apps are installed in the background. OEMs can also work with software developers to target OEM devices for apps to appear exclusively on, based on registry keys.
Change history for Customize	Review the timeline of Customization topics that have been created, updated, or deleted.

Audience

This section of the partner documentation is intended for Original Equipment Manufacturers (OEMs), Original Design Manufacturers (ODMs), Independent Hardware Vendors (IHVs), system builders, mobile operators, and IT administrators.

If you have purchased a Windows 10 device and would like to learn more about using its features, please see Microsoft's Windows support site online at https://support.microsoft.com/products/windows?os=windows-10 .

Manufacture

Article • 06/11/2024

Purpose

Use the manufacturing tools to deploy your customizations to new Windows 11 devices. Learn how to:

- Combine your customizations, plus languages, drivers, apps and more, into new Windows images.
- Modify these images either from a manufacturing mode for a familiar Windows experience, or from a command line for quicker changes that can be automated and scripted.
- Install images onto new devices. Choose whether to use compression to balance disk space verses device performance. Use flashing tools to speed up the final manufacturing processes
- Capture your customizations into the recovery tools, helping your customers get back up to speed quickly.

Expand table

Topic	Description
OEM deployment of Windows 11 for desktop editions	This guide is intended for OEMs, and applies to Windows 11 for desktop editions (Home, Pro, Enterprise, and Education). IT professionals using this guide should have prior knowledge of Windows basic administration and troubleshooting.
What's new in Windows manufacturing	Learn what new features are available.
Desktop manufacturing	Technical reference for Windows 11 for desktop editions
IoT Enterprise manufacturing	Technical reference for Windows IoT Enterprise
IoT Core manufacturing	Technical reference for Windows 10 IoT Core

Service

Article • 11/05/2020

Purpose

Update, refresh, or refurbish your after-market devices.

Goal	Resources
Updating the operating system, drivers and packages	Desktop: Upgrade Editions Drivers: Manage Driver Distribution in Windows 10 IoT Core device updates
Refurbishing devices	Desktop: Bare metal reset/recovery: create recovery media while deploying new devices Desktop: Bare metal reset/recovery: enable your users to create recovery media Desktop: Reset a device

Test for performance and compatibility

Article • 03/03/2022

We provide an array of tools and technology to help you test performance, compatibility, reliability, and functionality of the hardware and software you create to run with Windows 10. The Windows Hardware Lab Kit (HLK), the Windows Performance Tools (WPT) and the Windows Assessment Toolkit appeal to a wide audience that includes driver developers, hardware manufacturers, system builders, and general application developers.

About the tools

- Windows Hardware Lab Kit – The Windows HLK is a test framework used to test reliability and compatibility of hardware devices used with Windows 10.
- Windows Performance Toolkit – WPT helps you measure, analyze and troubleshoot performance to create fast, fluid, and efficient systems.
- Windows Assessment Toolkit – The assessment toolkit provides additional tools and tests that help you determine the quality of a running operating system or a set of components with regard to performance, reliability, and functionality.
- Windows Mixed Reality HMD Exerciser Kit - The Windows Mixed Reality Head Mounted Display (MR HMD) Exerciser Kit is a hardware and software package that automates the validation of MR HMDs and PCs.

Tool downloads

Download the Windows HLK	**Download the Windows ADK**
Previous versions of the kit are also available on the HLK download page.	The Windows Assessment and Deployment Kit (ADK) for Windows 10 includes WPT and the assessment toolkit as well as several useful deployment tools.

Step-by-step guidance

- HLK Getting Started Guide
- Windows Performance step-by-step guides
- Windows Assessment Console step-by-step guide
- Windows MR HMD Exerciser Kit Quick Start Guide

Related resources

- Dashboard services – The hardware dashboard is used to submit HLK tests and certify devices and apps, to submit and distribute drivers, and to manage your partner account.
- Windows Engineering Guides (WEGs) - WEGs provide partners with guidance in delivering Windows-based computers and devices to customers.
- Windows Compatible Products List– This program provides you with the opportunity to list your product on Microsoft's Certified Products List after passing the compatibility and reliability tests in the HLK.

Developer audience

This section of partner documentation is intended for OEMs, ODMs, and IT professionals.